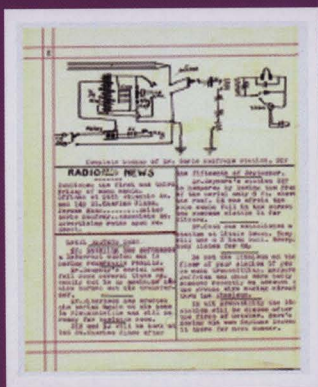
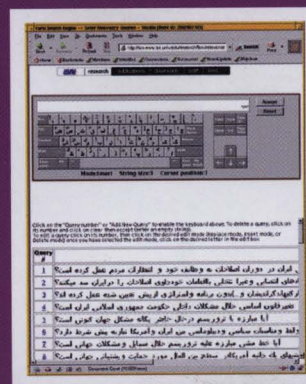


Proceedings

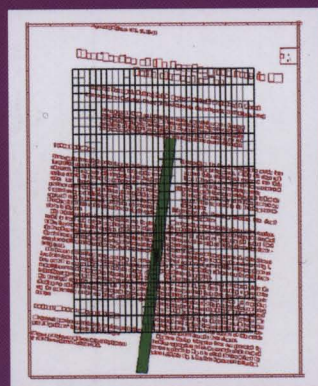
2003 Symposium on Document Image Understanding Technology



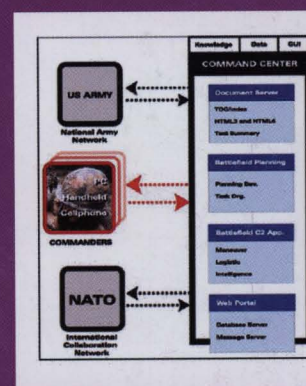
Degraded Documents



Multilingual Documents



Page Structure



Multimedia

April 9-11, 2003
Marriott Hotel
Greenbelt, Maryland

Proceedings
SDIUT03
The 2003 Symposium on
Document Image Understanding Technology

Greenbelt Marriott Hotel
Greenbelt, Maryland
April 9-11, 2003



SDIUT '03



Sponsored by: The United States Department of Defense
Organized by: The Laboratory for Language And Media Processing
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

A limited number of additional copies of these proceedings are available for \$60 from:
University of Maryland
Institute for Advanced Computer Studies
College Park, MD 20742
Phone (301) 405-6444
Fax: (301) 314-9115
Email: sdiut03@lamp.cfar.umd.edu

Table of Contents

MESSAGE FROM THE ORGANIZERS7

KEYNOTE SPEAKERS

DOCUMENT PROCESSING AND UNDERSTANDING: AN INTEGRATED APPROACH..... 11
Patrice Simard, Microsoft Research

OVERVIEW OF THE QUESTIONED DOCUMENT UNIT, FBI LABORATORY..... 13
Gabriel Watts, Federal Bureau of Investigation

SESSION 1 PAGE STRUCTURE 1

ASSURING HIGH-ACCURACY DOCUMENT UNDERSTANDING: RETARGETING, SCALING
UP, AND ADAPTING.....17
*Henry Baird, Kris Papat, Thomas Breuel, Prateek Sarkar and Daniel Lopresti,
Palo Alto Research Center*

AUTOMATED DATA EXTRACTION FROM STRUCTURED DOCUMENTS.....31
Janusz Wnek, Science Applications International Corporation

AUTOMATED LOGO DETECTION AND RECOGNITION.....37
Tom Drayer and Ken Cantwell, Department of Defense

SESSION 2 MULTILINGUAL DOCUMENTS

FARSI SEARCHING AND DISPLAY TECHNOLOGIES.....41
*Kazem Taghva, Ron Young, Jeff Coombs, Ray Pereda, Russell Beckley and
Mohammad Sadeh, University of Nevada, Las Vegas*

PORTING THE BBN BYBLOS OCR SYSTEM TO NEW LANGUAGES.....47
*Prem Natarajan, Michael Decerbo, Tom Keller, Rich Schwartz
and John Makhoul, BBN Technologies*

SEGMENTING AND TAGGING STRUCTURED CONTENT53
Huanfeng Ma, Burcu Karagol-Ayan and David Doermann, University of Maryland

SESSION 3 HANDWRITING

| | |
|---|-----------|
| A SYSTEM FOR HANDWRITING MATCHING AND RECOGNITION | 67 |
| <i>Sargur N. Srihari, Bin Zhang, Catalin Tomai, Sangjik Lee, Zhixin Shi and Yong-Chul Shin, CEDAR University of Buffalo</i> | |
| INDEXING OF HANDWRITTEN HISTORICAL DOCUMENTS – RECENT PROGRESS | 77 |
| <i>R. Manmatha and Toni M. Rath, University of Massachusetts, Amherst</i> | |
| DOCUMENT CATEGORIZATION USING LATENT SEMANTIC INDEXING | 87 |
| <i>Anthony Zukas and Robert Price, Science Applications International Corporation</i> | |
| PARSING FREEFORM HANDWRITTEN NOTES ON THE TABLET PC | 93 |
| <i>Michael Shilman, Microsoft Research</i> | |

SESSION 4 DEGRADED DOCUMENTS

| | |
|--|------------|
| PROCESSING NOISY DOCUMENTS | 97 |
| <i>Yefeng Zheng, Huiping Li and David Doermann, University of Maryland, College Park</i> | |
| SUMMARIZING NOISY DOCUMENTS | 111 |
| <i>Hongyan Jing, Daniel Lopresti and Chilin Shih, Bell-Laboratories, Lucent Technologies, Inc.</i> | |
| ROUGH AND DEGRADED DOCUMENT INTERPRETATION BY PERCEPTUAL ORGANIZATION | 121 |
| <i>Eric Saund, David Fleet, James V. Mahoney and Daniel Larnier Palo Alto Research Center</i> | |
| OCR ACCURACY PREDICTION AS A SCRIPT IDENTIFICATION PROBLEM | 135 |
| <i>Vitaly Ablavsky, Joshua Pollak, Magnus Snorrason, and Mark R. Stevens, Charles River Analytics Inc.</i> | |

SESSION 5 OCR AND OCR CORRECTION

| | |
|---|------------|
| A SURVEY OF RETRIEVAL STRATEGIES FOR OCR TEXT COLLECTIONS | 145 |
| <i>Steven Beitzel, Eric Jensen and David Grossman, Illinois Institute of Technology</i> | |
| OCR ACCURACY AND RETRIEVABILITY OF POST-PROCESSED DOCUMENTS | 153 |
| <i>Tom Nartker, Kazem Taghva and Julie Borsack, University of Nevada, Las Vegas</i> | |

| | |
|---|-----|
| VARYING EFFECTS OF IMAGE IMPROVEMENT METHODS ON OCR ACCURACY | 159 |
| <i>Kristen Summers, Vredenburg</i> | |

| | |
|---|-----|
| OCR CORRECTION USING HISTORICAL RELATIONSHIPS FROM VERIFIED TEXT IN BIOMEDICAL CITATIONS | 171 |
| <i>Susan Hauser, Tehseen Sabir, George Thoma, National Library of Medicine</i> | |

SESSION 6 DOCUMENT ANALYSIS RESOURCES

| | |
|--|-----|
| BALANCED QUERY METHODS FOR IMPROVING OCR-BASED RETRIEVAL | 181 |
| <i>Kareem Darwish and Douglas Oard, University of Maryland, College Park</i> | |

| | |
|--|-----|
| CREATION OF MULTI-LINGUAL DATA RESOURCES AND EVALUATION TOOL FOR OCR | 189 |
| <i>S. Setlur, S. Kompalli, R. Vemulapati, V. Govindaraju, State University of New York/Cedar Buffalo</i> | |

| | |
|--|-----|
| MULTILINGUAL OCR GROUND TRUTH FROM PRINTED AND WEB SOURCES | 197 |
| <i>Mark Turner, Yuliya Katsnelson, and Kristen Summers, Vredenburg, Incorporated</i> | |

| | |
|--|-----|
| GROUND TRUTH DATA FOR DOCUMENT IMAGE ANALYSIS | 199 |
| <i>Glenn Ford, and George R. Thoma, National Library of Medicine</i> | |

SESSION 7 PAGE STRUCTURE 2

| | |
|--|-----|
| HIGH PERFORMANCE DOCUMENT LAYOUT ANALYSIS | 209 |
| <i>Thomas M. Breuel, Palo Alto Research Center</i> | |

| | |
|--|-----|
| AUTOMATED LAYOUT RECOGNITION | 219 |
| <i>Lynn Golebiowski, Booz Allen Hamilton</i> | |

| | |
|---|-----|
| LAYOUT BASED CLUSTERING USING NORMALIZED MATCHING MUTUAL INFORMATION FOR BINARY DATA | 229 |
| <i>Alan Sakakihara, Booz Allen Hamilton</i> | |

| | |
|---|-----|
| AUTOMATIC FORMS PROCESSING IN THE NIST FORMS DATABASE DOCUMENT IMAGE UNDERSTANDING TECHNOLOGY 2003 | 239 |
| <i>Carson Cumbee, Department of Defense</i> | |

| | |
|---|-----|
| AMPLIFYING ACCURACY THROUGH STYLE-CONSISTENCY | 245 |
| <i>Prateek Sarkar, Thomas Breuel, Palo Alto Research Center</i> | |

SESSION 8 MULTIMEDIA

| | |
|---|------------|
| FORM ANALYSIS WITH THE NONDETERMINISTIC AGENT SYSTEM (NDAS)..... | 253 |
| <i>Thomas Henderson and Lavanya Swaminathan, University of Utah</i> | |
| METRICS FOR EVALUATING THE PERFORMANCE OF VIDEO TEXT RECOGNITION SYSTEMS..... | 259 |
| <i>Gregory K. Myers, SRI International</i> | |
| UNIVERSAL DOCUMENT MANAGEMENT SYSTEM FOR THE MOBILE WARRIOR..... | 265 |
| <i>H. Alam, R. Hartono, Fuad Rahman, Y. Tarnikova, T. Tjahjadi, and C. Wilcox, BCL Technologies</i> | |

ADDITIONAL MATERIAL

DEMOS AND ABSTRACTS

| | |
|--|------------|
| THE CAMERA FRAMEWORK FOR BUILDING CUSTOM RECOGNITION SYSTEMS | 275 |
| <i>Michael Droettboom, Karl MacMillan and Ichiro Fujinaga, The Johns Hopkins University</i> | |
| 3D METHODS TO AID HANDWRITING ANALYSIS AND OCR | 287 |
| <i>Anshuman Razdan, John Femiani, Jeremy Rowe, Arizona State University</i> | |
| AUTOMATED READING OF FREE-FORM HANDWRITING IN IMAGES, THE PAST AND ONE PROPOSED FUTURE..... | 289 |
| <i>Joanna Fancy, HigherGlyphics</i> | |
| THE VIDEO SPECTRAL COMPARATOR 2000HR..... | 291 |
| <i>Greggory Mokrzycki, Federal Bureau of Investigation</i> | |
| DOCUMENT LAYOUT ANALYSIS | 293 |
| <i>Thomas Breuel, Palo Alto Research Center</i> | |
| HIGH VIEW DOCUMENT IMAGE MANAGEMENT TOOL..... | 295 |
| <i>Mark Turner, Vredenburg</i> | |
| SCANSOFT ASIAN LANGUAGE OCR CAPABILITY | 307 |
| <i>Tom D'Errico, ScanSoft</i> | |
| GROUNDTRUTH IMAGE GENERATION FROM ELECTRONIC TEXT (DEMONSTRATION)..... | 309 |
| <i>David Doermann and Gang Zi, University of Maryland, College Park</i> | |

A GENERATIVE PROBABILISTIC OCR MODEL.....313
*Okan Kolak, Philip Resnik and William Byrne, University of Maryland, College Park,
The Johns Hopkins University*

GOVERNMENT APPLICATIONS

**OCR FOR COLLECTION, MANAGEMENT, AND RETRIEVAL OF DOCUMENTS:
DEVELOPMENT AND TRIAL OF A DOCUMENTATION EXPLOITATION SUITE321**
Luis Hernandez and Christian Schlesiger, Army Research Laboratory

TRANSITIONING EXPERIMENTAL HMM OCR: FROM LAB TO FIELD.....327
Christian Schlesiger, Luis Hernandez, and Michael Lee, Army Research Laboratory

THE EFFECTS OF DOCUMENT ANALYSIS ON AUTOMATIC CONTENT EXTRACTION331
Jonathan K. Davis, Department of Defense

AN AUTOMATION TOOL FOR THE DETECTION OF SENSITIVE INFORMATION339
Gary DeWitt, Department of Energy

THE DECLASSIFICATION CHALLENGE: CAN TECHNOLOGY MAKE A DIFFERENCE?341
Richard Warshaw, Central Intelligence Agency

VACE ADVANCED R&D PROGRAM.....343
John Prange, Advanced Research and Development Activity

AUTHOR INDEX355

Message from the Organizers

Welcome to the fifth bi-annual symposium on Document Image Understanding Technologies (SDIUT). This year's meeting in Greenbelt closes in on a decade of meetings, which have attempted to bring together researchers and research sponsors from government, academia and industry to explore trends in document image analysis research and to identify the areas of primary interest in the field. Over the years, we have seen a great deal of progress in all aspects of document analysis, and in particular, with the types of applications. This year's keynote talks are representative. On one hand, we hear about how document analysis being tailored to the masses at Microsoft, and on the other, how ultimately, we can apply document analysis to the very specific goals of the Questioned Documents Unit at the Federal Bureau of Investigation.

This year's symposium will host over 25 technical and government talks, and a number of demo, exhibits and posters. These proceedings provide insight into the details of the technical presentations, and overviews of the material presented by government speakers and keynote presenters, but are not a substitute for the interaction available with others in the field. For those of you who are participating in this year's symposium, we hope you enjoy the presentations and take an active part in the discussions, which develop throughout the meeting. For those who are reading these Proceedings, we encourage you to follow-up with the authors and start dialogs about the problems you are trying to address. We feel that this symposium is best viewed as a catalyst for more in-depth offline discussion.

The organization of this meeting was handled in part by the staff of the University of Maryland Institute for Advanced Computer Studies (UMIACS); in particular, we thank Yang Wang for his support of the WWW site, and Christopher McCarthy for his graphics design expertise.

Finally, we would once again, like to thank Ms. Denise Best dedication to organizing all aspects of this meeting. She is truly the reason why things keep moving in the right direction.

Thank you for your participation.

The SDIUT '03 Organizers and Sponsors.

Keynote Speakers

KEYNOTE SPEAKER

Document Processing and Understanding: An Integrated Approach

Patrice Simard

Microsoft Research

Document Processing & Understanding (DPU)

One Microsoft Way, 113/3354, Redmond WA 98052, USA

ABSTRACT

Document manipulation is a major business for Microsoft. Our research focuses on developing new technologies to understand, recover, and generate documents in both printed form and electronic ink. This talk will focus on our experiences with parsing freeform handwritten notes, handwriting recognition, printed document layout analysis, and annotations. These problems are interrelated, and solving them together yields an economy of scale. Examples and solutions will be illustrated on a Tablet PC.

BIOGRAPHICAL SKETCH

Patrice Simard manages the Document Processing & Understanding (DPU) group at Microsoft Research. His team is responsible for developing new technologies related to documents. Examples include recovering electronic documents from their printed version (reviving dead bits), converting free-form handwritten notes to structured electronic documents, understanding documents for compression and re-purposing, and automatically generating document layouts.

Patrice has filed 27 patents and is the author of over 40 scientific publications in the fields of Machine learning, Vision, and Signal Processing. Before joining Microsoft, Patrice worked in research with AT&T Laboratories from 1991 to 1998. He received his bachelor's degree in Electrical Engineering (1986) from l'Université de Montréal, Canada and his Ph.D. in Computer Science (1991) from the University of Rochester, NY. His scientific interests include algorithms, compression, learning, and generalization.

KEYNOTE SPEAKER

Overview of the Questioned Document Unit, FBI Laboratory

Gabriel Watts

Forensic Document Examiner, Questioned Documents Unit

FBI Laboratory

2501 Investigation Parkway

Quantico, VA 22135

ABSTRACT

The Questioned Documents Unit (QDU), FBI Laboratory, is tasked with the forensic examination of documents submitted from various federal, state, and local law enforcement agencies. Examples of documentary evidence encountered by examiners include charred debris from the September 11 terrorist attacks, shredded paper from the Enron investigation, typewritten letters and typewriters from the UNABOMER, fraudulent sports memorabilia, and letters and envelopes used in the anthrax attacks. The QDU considers documents to be any physical substrate, which may contain information of investigative value. That information may come in the form of handwriting, print processes, typewriting, or physical characteristics such as torn edges, striation marks or reflective properties. Scientific techniques and specialized equipment are used to uncover and detect latent indented writing, invisible or obliterated inks, minute defects in typewriting, and unique patterns in handwriting characteristics.

Due to the extensive scope and volume of cases worked by the FBI document examiners, it is necessary to maintain informational databases that are utilized to cross check evidence in seemingly unrelated crimes. These databases, which include the Forensic Information System for Handwriting, Bank Robbery Note File, Anonymous Letter File, and the National Fraudulent Check File have been an invaluable tool for associating evidence between different cases. This presentation will show how the Questioned Documents Unit combines rudimentary forensic science with state of the art technology to question, detect, and uncover consequential evidence that would have been otherwise overlooked.

BIOGRAPHICAL SKETCH

Gabriel Watts, a graduate of Old Dominion University, first became interested in forensic science in 1994 while working as an intern in the Counterfeit Division of the United States Secret Service. He began with the FBI in 1997 and completed a full time 2-year training program in forensic document examination. He has conducted hundreds of examinations in cases ranging from bank robberies to bomb threats and testifies as an expert witness for the FBI in the United States and abroad. He is a member of the questioned document section of the Mid Atlantic Association of Forensic Scientists (MAAFS).

Page Structure 1

Assuring High-Accuracy Document Understanding: Retargeting, Scaling Up, and Adapting

Henry Baird, Kris Papat, Thomas Breuel, Prateek Sarkar, and Daniel Lopresti
Palo Alto Research Center
3333 Coyote Hill Road, Palo Alto, CA 94304

Abstract

*No existing document-image understanding technology, whether experimental or commercially available, can guarantee high accuracy across the full range of documents of interest to government-agency users. Research at PARC has focused for more than ten years on relieving this critical bottleneck to automatic analysis of the contents of paper-based documents, FAXes, etc. PARC has made significant progress — documented in dozens of publications and patents, and embodied in experimental software tools — towards this goal: we possess **document image decoding (DID)** technology that achieves high accuracy on images of documents printed in a potentially wide variety of writing systems and typefaces, unusual page layout styles, and severely degraded image quality. Our principal method of attack has been **retargeting**: that is, our technology is designed to be trainable, i.e. customized to the characteristics of individual documents or sets of similar documents. In recent years we have reduced the effort of manual DID training significantly. In this paper we propose **scaling up** the DID methodology by the use of massively parallel recognition using ensembles of automatically pre-trained DID decoders: this promises to reduce further the need for document-specific training. We have also made recent progress towards **adapting**, in which recognizers, without any manual training, adjust their models to fit the document at hand: this offers hope that manual training can someday be reduced to zero. Finally, we have extended DID methods to handle gray-level images. All of these R&D projects are ripe for accelerated extension, experimentation, and application to government-agency needs.*

1 Introduction

Most of the challenges faced by many US government agencies in automating the capture, understanding, and reuse of scanned paper documents are

illustrated in an extreme form by the enormous unmet technology needs of the intelligence community. When a crisis occurs in a country/language/culture that is not supported by modern information technology infrastructures, much crucial data that are captured, intercepted, or needed for reference are available only as printed text on paper. The high volume, wide variety, and often low quality of these printed documents pose severe challenges to efforts rapidly to 'ramp-up' intelligence analysis by converting the text found in these documents to computer-legible encoded form (e.g. XML, Unicode, ASCII).

In particular, high-accuracy OCR systems may not exist for obscure languages and writing systems due to the lack of commercial incentives to develop them. Also, many of the documents, when scanned, may yield images of such low quality that conventional OCR systems fail almost completely. Although modern OCR systems often use language-specific models to improve recognition, it may often be the case in intelligence applications, for best results, that specialized technical documents require 'language models' customized to their domain of discourse, not merely to the language. Essentially the same challenges — specialized domains, low-quality images, complex nonstandard page layouts — arise in many governmental non-intelligence applications.

2 Retargeting to Individual Documents

PARC's proprietary experimental document recognition technology, called *document image decoding (DID)*, is able to convert textual data found in scanned images of machine-printed documents into digitally encoded form (e.g. XML, Unicode, ASCII). This DID technology is the fruit of over a decade's basic research at PARC into high-accuracy retargetable document image understanding algorithms. Software prototypes implementing many of the DID algorithms are presently running in our laboratory. Government assistance is essential to extend the

DID tools and demonstrate their success on difficult cases of special value to the intelligence community and other government agencies.

Key advantages of the DID technology to the intelligence community:

- **Retargetable to specific documents for a large reduction in error rate.** Experiments at PARC have shown that when the models are a good fit to the document, DID decoding yields an order of magnitude fewer errors than conventional recognition methods [1]. *The advantage to the intelligence community is that they can reliably convert high-value, urgently needed documents, where recognition errors must be kept low and there is no time for extensive manual correction of OCR results.*
- **Models are trainable from examples.** Typeface and image quality models are semi-automatically trainable given samples of page images and ground-truth text. The language model is trainable fully automatically given a (preferably large) corpus of correctly encoded text in the language and domain of discourse. *The advantage to the intelligence community is that, in order to handle a new language, most of what is needed is merely to provide sample data for training; it is no longer necessary to develop language-specific image processing algorithms and software. Thus, many months or even years of delay are avoided, and a rapid ramp-up of intelligence capability becomes possible.*
- **Segmentation-free.** DID recognition results are guaranteed to be the best possible (with respect to the models) over all segmentations of text lines and words into characters. *The advantage to the intelligence community is that it is not necessary to build a separate segmentation stage of processing for each new language or writing system.*
- **Formally provably optimal recognition.** DID theory guarantees, in a precise, realistic, and useful sense, that its interpretation is the best possible simultaneously over the cross product of four document-specific models: glyph templates, image quality, language/domain of discourse, and page layout. The proofs of these claims depend on certain technical assumptions, including that the Markov hypothesis holds. *The advantage to the intelligence community is that DID recognition accuracy tends to be high even if one or more of the models is less than perfect. This occurs because the DID algorithms combine all*

three models, at decoding time, in a single, unified computation. This is in contrast to almost all competing methods which rely on heuristics (lacking formal guarantees) to combine the results of separate stages of computation (often themselves heuristic), and thus are unable to achieve global optimality. In practical terms, these formal guarantees not only make DID more accurate, but they also make it tolerant of less-than-ideal training data and easier to extend to new applications.

- **Generative side-bearing typesetting model.** One of the unique features of the DID framework is its incorporation of an explicit side-bearing model of typesetting (e.g. a page, text-block, and/or text-line layout model) as a probabilistic generative model. Decoding under this model is equivalent to picking the best (optimal) match, to the given image, among all possible layouts that can be generated by the model. Thus every randomized generative run of the DID model will produce a line of text; by contrast, other non-character-based HMMs proposed in the literature (e.g. by BBN) may produce gibberish images not containing any recognizable characters. We believe this stronger constraint gives DID an edge in text recognition especially for low quality images.

2.1 The Theory and Technology of Document Image Decoding

The Document Image Decoding (DID) approach to machine-printed character recognition, invented at PARC and extended and improved over a decade, finds a sequence of characters that best explains an observed document image in terms of explicit models of printing, scanning, and language. It is based on a communication-systems interpretation that views the generation of a printed page in terms of transitioning through a probabilistic finite-state machine, or Markov model. In this model of the image source, a mark or template (typically corresponding to an individual character) is printed upon every state transition, and the current printing location on the page is advanced by the corresponding 'set-width' (actually, a 2-D displacement vector). The observed image is viewed as a possibly degraded version of the ideal image produced by the Markov model.

The essential task performed in DID is to work backwards from the observed image to reason about what path must have been followed through the Markov source, and what the degradation must have been, to produce the image. A variant of dynamic programming was used to accomplish this in the original work on DID [13], under the assumption

that the Markov source is amenable to causal processing via scheduling. In more recent work ([18], [16], [14], [11], [12], [15], [19], [20], [17], [2]), this assumption is maintained but its significance somewhat obviated by an independent restriction to individual text lines which are one-dimensional and hence trivially schedulable: this line-by-line decoder is fully implemented.

Several models must be specified for DID to work on any particular document. Three of these models are learned from training data: (1) "typefaces," the shapes and identities of the templates printed by the Markov source, (2) "image quality," the manner in which the ideal image is corrupted on the way to observation, and (3) "language," a prior description of which recognized strings are valid in the given language. One remaining DID model is structural: these are the states and transitions in the Markov model itself; in the special case of an isolated text-line decoder this is a simple, fixed three-state Markov model (in other more complicated cases, this model would be specified manually).

Given training images, learning typefaces becomes essentially a character segmentation problem, which we have addressed by a graph-theoretic independent-set formulation [16], an iterative procedure [14], and most recently as an instance of the expectation-maximization algorithm [14] [11].

The second modeling problem, learning image quality (or, degradation parameters) was initially addressed using a binary asymmetric pixel-flip model [13], then generalized to allow different pixel-flip probabilities depending on position within the template [12], and finally to grayscale observations with the possibility of spatial dependence in the deterministic component of the degradation [18]. For the third class of models needed by DID, the language model, a unigram (simple letter frequency) model was initially used [13]; this was then extended to fixed [15] and variable [19] character N-grams (statistical characterization in terms of groups of N adjacent characters). When integrating an N-gram language model into Viterbi template-matching search, conventional dynamic programming is no longer feasible due to a potentially exponential blow-up in the search space. In response to this problem we invented a procedure to find an optimal path that is, on average, far faster [19] and compared it with an approximate (sub-optimal) technique that is often faster still [20].

Large reductions (up to a factor of forty) in the concrete as well as asymptotic time complexity of DID have been realized without compromising optimality by iterative decoding using heuristic upper bounds and segmental dynamic programming [17], and judicious subsampling [2]. A recent summary of

these potentially practically important performance improvements appeared as [3].

A strategic overview, from the point of view of a technically sophisticated end-user, of the motivation, history, technical advances, and potential applications of DID is given in [1].

2.2 Ways in Which DID Is Not Restricted to Particular Languages & Writing Systems

In the DID framework, the Markov model of the generation of the ideal image makes only minimal assumptions about typefaces and/or writing systems. Glyph templates can correspond to characters, parts of characters, or groups of characters. Ligatures can be handled by expanding the font to include them as separate templates, and diacritical marks can be handled in the same way, or else by adding only the diacritic to the font, and making provisions within the Markov model to place it in the appropriate position relative to the character it modifies. The number of templates in a font is unbounded in principle, although execution time of both training and decoding is increased (linearly) as the number of templates grows. The displacement ('set-width') vector associated with the printing of each template is, in the most general setting, a two-dimensional quantity: this allows DID Markov models to accommodate right-to-left (e.g. Semitic) and top-to-bottom (e.g. East Asian) writing systems, in principle, although only left-to-right writing systems have been implemented to date. An advantage of having a probabilistic generative model for text-line layout is that such a model can be trained automatically from incomplete data by means of well studied algorithms such as Expectation Maximization (EM). In particular, we possess experimental software for which training requires only text-line images and their transcriptions (encoded ground-truth text) as input, and infers and iteratively refines the DID models and parameters (for template shape, degradation, and text-line layout) while making progressively better guesses about the true positions of the characters along the text line. This algorithm is initialized with a guess at the model parameters: this guess does not have to be exact, as has been shown in laboratory experiments in which the training algorithms have improved recognition performance even when initialized with fonts dissimilar to the ones occurring on the page. The primary advantage of such a procedure is that it does not require specification of character-level alignment in the ground truth, thereby making ground truth collection simpler and less costly.

2.3 Ways in Which DID May be Restricted to Particular Languages & Writing Systems

DID assumes that character artwork (glyphs) takes the form of "templates," that is, each glyph is modeled by one or more fixed-size, rigid images. Therefore DID cannot be applied to freeform scripts such as handwriting where the shape of a character changes depending on context. Also, the DID framework requires that neighboring character templates do not "overlap": more precisely, glyph supports must be disjoint (share no pixels), where the support of a glyph is the set of pixels whose values (write-black, write-white, boundary, etc) determine the identity of the glyph. But this restriction is not as limiting as it may at first seem, since supports need not be rectangular (they may be any connected set of pixels), and the supports of neighboring characters may extend above or below one another as well as to the left and right of one another. While "compound characters" (e.g. accented characters such as , digraphs such as) can be specified as special glyphs, scripts that involve extensive shape-interference between characters cannot be handled without more research and development. Our current implementation of DID tools handles baselines that are nominally horizontal straight lines (although they may be 'skewed' and may even curve quite a bit). Within a text line, symbols are expected to be printed strictly left-to-right: other alternatives (right-to-left, top-to-bottom, and bottom-to-top) are permitted with no change in the theory and can be accomplished by straightforward software engineering. Our current language models are variable character n-gram based (where n is usually at most 4): to be applicable to new languages for best results a large quantity of correctly encoded 'ground-truth' training data is required to ensure that the vast majority of cases likely to occur in the document to be decoded will have appeared in the training corpus.

2.4 The Current Software Implementation of the DID Tools

2.4.1 Fonts & image quality models

Our current best tool for training implements an Expectation Maximization (EM) training algorithm which iteratively refines the parameters for models of template shape, image degradation, and text-line layout on the basis of text images, text-line locations, and text transcriptions (so that character alignment locations need not be specified). The code-base currently comprises of more than 40,000 lines of ANSI C, and several thousand lines of glueware in shell-scripts and makefiles. It has been suc-

cessfully run on single font (typeface and typesize) English text demonstrating improved OCR performance with the adapted models. The code-base requires testing on large datasets to systematically document speed and performance in a wide variety of cases, and some engineering modifications to gracefully handle unexpected situations.

The version of decoding software most suitable to serve as a starting point for a deployable version is written in C. It incorporates the speed enhancements described in references [17] and [2], but does not yet incorporate a language model. It consists of 9,000 lines of code and has some dependencies on a 350,000-line image processing library. It has been tested on images of dozens of lines of text.

2.4.2 Language models

The DID language model is a character N-gram, with automatically varying N, and has a built-in smoothing mechanism, both included to handle training sparse data. It also has provision for generating upper-bounds for use with the iterative search algorithm described in [19] and [20]. It consists of about 4,000 lines of C++ code, and is documented at the interface level with a Unix-style manua page and two example calling programs. The language model has been tested on recognition in low-resolution grayscale text images; training has been carried out using as much as 20 million characters, and decoding using Stack and Iterative Search have each been tested on scores of text lines. It is not currently optimized for speed, and accessing the language model probability estimates and upper bounds currently accounts for a substantial fraction of the decoding time in the simulations we have done to date. The primary data structure in the language model is a trie. The speed could be substantially improved by replacing the linked list structure that holds the children of each interior node in the trie with a hash table, especially at shallower depths in the trie.

With government support we are prepared to extend, prove, and apply these tools as follows:

1. Extension of image recognition tools to include language model.
2. Extension of all DID tools to 16-bit Unicode, multifont, & multistyle.
3. Benchmark DID tools on non-English text.
4. Exercise image recognition tools on a connected writing system.
5. Characterize the applicability of DID tools to an underserved language/script.
6. Deliver DID software tools to government.

2.5 Training on Severely Degraded Text-Line Images

We have recently shown [22] that DID supervised training algorithms can achieve high accuracy with low manual effort even under conditions of severe image degradation in both training and test data. This is the result of improvements in DID training of character template, set-width, and channel (noise) models. Large-scale experimental trials, using synthetically degraded images of text, have established two new and practically important advantages of DID algorithms:

1. high accuracy (>99% characters correct) in decoding using models trained on even **severely degraded images** from the same distribution; and
2. greatly improved accuracy (<1/10 the error rate) **across a wide range of image degradations** compared to untrained (idealized) models.

Here are some examples of degradations for which these claims hold true:

gRv13 UF4dg eJWCc F9jgz

This ability to train reliably on low-quality images that suffer from massive fragmentation and merging of characters, without the need for manual segmentation and labeling of character images, significantly reduces the manual effort of DID training.

These algorithms eliminate the need for segmenting page images into isolated glyphs before training on them. All they require is presegmentation into *images of entire lines of text* presented in the same order as the ground-truth lines of encoded text. The algorithms automatically align glyph images with their corresponding encoded characters in the ground truth. The training process is robust even when the initial choices of templates differ significantly in image quality from the final trained results. Robustness in the face of such differences reduces the manual effort, skill-level, and time required — and thus reduces training costs.

We have run large-scale experimental trials in which these algorithms proved able to learn DID models of high quality (that is, enabling recognition with accuracy greater than 99%) even when the training and test images are of low quality. The trials were carried out using pseudo-randomly degraded images permitting systematic mapping of the domain of competency of the training and decoding algorithms.

3 Scaling-Up to Recognize Many Types of Documents

Commercial OCR packages work well on average, but offer no guarantee that any particular word or sentence will be correctly recognized. In recent years, the use of ensembles — collections of different OCR engines working in parallel on the same input, with their outputs combined by some form of weighting voting — has been proposed as a way to boost overall accuracy. While ensembles have proven to be effective for this purpose, the risk remains strong that all or most of the OCR engines will fail, together, on any specific piece of problematic text, thereby causing the ensemble to likewise fail on that text. Among the factors influencing the effectiveness of ensembles is the voting method, the individual accuracies of the component OCR engines, and how "different" the OCR engines are in the specifics of when and how they fail. Ideally, for any piece of text at least one OCR engine in the ensemble will recognize it correctly, and the weighted voting will pass that correct recognition on to the output.

If sufficiently many OCR engines are used in such an ensemble, correct recognition is essentially assured. To achieve this, there must be a means of tuning each OCR engine in the ensemble to a different type of text image — a different type of degradation, a different font, a different language, etc. Unfortunately, existing commercial OCR engines do not provide sufficient control over their inner workings to allow such tuning. In contrast, the Document Image Decoding (DID) approach to text recognition is richly tunable, owing to its firm grounding in a unifying probabilistic framework. Each aspect of the text to be recognized is modeled explicitly: the layout, font, language, and degradation type and severity. In abstract terms, any instance of these factors correspond to a point in a high-dimensional parameter space that characterizes a particular DID OCR engine. By choosing a set of parameterizations that *cover all likely regions of parameter space*, an ensemble can be built that has the desired property that at least one will "get it right" for any input.

One of the key properties of such an ensemble is its *scalability*. When stronger assurance/accuracy is required, a larger ensemble can be built. Since the dimensionality of the parameter space is quite high, it is likely that the ensembles in question will consist of hundreds or perhaps thousands of DID engines running in parallel. The precise number will depend on requirements and the available resources, but the key point is that the high degree of parameterizability of DID makes such an approach possible and promising, advancing the state of what is possible in recognition accuracy and robustness.

4 Adapting to the Document

PARC can also offer rapidly to advance the state of the art of adaptive recognition systems. An adaptive system estimates the parameters of the DID models best fitted to the document at hand based on evidence extracted automatically on the fly from the document itself, and then uses these document-specific parameters to decode the rest of the document, or to re-decode the whole document. This promises two advantages over massively parallel scalable tuned recognizers:

1. higher accuracy: the adapted models can converge to a closer fit to the document at hand, and so lead to higher accuracy; and
2. higher processing speed: fewer tuned recognizers could be used, both initially when deciding what range of model parameters are the best fit to the document, and later as parameter estimates converge.

It also promises to achieve higher accuracy than re-targetable recognition, in as much as the model parameters continue to be re-estimated, more and more accurately, after manual training has ended.

We have studied adaptive systems for several years, especially in the context of *isogenous* documents, which are documents generated from a stable set of parameters, but whose exact parameters are not known. Documents with this property are very common:

- **Handprinted characters** in a form-field often come from the same writer with the same writing device,
- **Printed characters** appearing in word are likely to share a common font,
- **Speech** segments in a directory assistance call may come from the same speaker through the same recording device and transmission channels,
- **Characters** on documents that undergo copying, faxing or scanning processes are all subjected through the same **degradation** mechanisms,
- objects in **aerial photographs** all share similar levels of illumination and viewing angle,
- Two articles/reports encoding the same information reflect the authors' "style" in the **choice of words and phrases**.

We have shown in laboratory experiments that recognition errors can be cut by 25-40% by exploiting style consistency. We invite government support for a two-year research program to improve on

these early successes and apply this new technology to problems of pressing importance to the government.

In all these situations modeling style, in the form of the statistical dependence among co-occurring patterns, aids in the process of pattern recognition/analysis. In laboratory experiments, our statistical models and classification algorithms have reduced OCR errors by 25-40%. With the rapid improvement of accessible computing resources, sophisticated models of context can be successfully applied to pattern recognition problems such as document analysis. Applications such as OCR and speech recognition have benefited immensely from modeling linguistic context. Exploiting style context appears to be an attractive next step.

Prateek Sarkar wrote a doctoral thesis on isogenous patterns, style consistency models, and style constrained classification, demonstrating that styles can be *learnt automatically* from training data. Thomas Breuel has developed techniques for adaptive classification based on hierarchical Bayesian methods.

The strengths of our methods lie in:

- Automatic learning without style-labeled data. This enables style-consistency to be exploited even without expensive manual "ground-truthing" of large data-sets. This is especially important when styles are difficult to enumerate, e.g., handwriting, speech, degradation.
- Adaptive systems that learn styles of previously unseen data (e.g., obscure fonts, unseen writers).
- Fast but optimal classification, under the assumptions of the statistical model.

We intend, under an agency sponsorship, to generate software that is broadly designed to apply to different kinds of data, and conduct large scale experiments on suitable data that may be provided by the sponsoring agency. The resulting software system and experimental findings will be made available to the sponsoring agency, while the investigators will benefit from the completed research software and publications.

In addition, we have re-examined a well-known technique in OCR, recognition by clustering followed by cryptanalysis, from a Bayesian perspective. The advantage of such techniques is that they are font-independent, but they appear not to have offered competitive performance with other pattern recognition techniques in the past. Our analysis suggests a novel approach to OCR that is based on modeling the sample distribution as a mixture of Gaussians. Results suggest that such an approach may combine

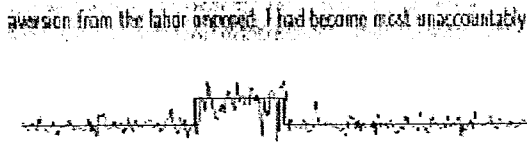


Figure 1: Top: example of a text line image corrupted by nonstationary additive white pseudorandom noise. Bottom: corresponding root-mean-square column intensity clearly shows nonstationary profile; the straight lines indicate the known standard deviation of the pseudorandom noise.

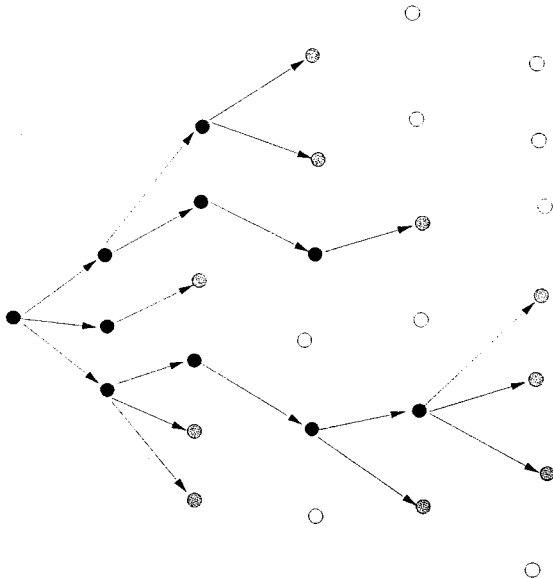


Figure 2: Stack algorithm search tree in the midst of decoding a line. The solid dots have been explored while the unfilled circles have not. The gray dots are the frontier nodes.

the advantages of cluster-based OCR with the performance of traditional classification algorithms.

Extending and generalizing our prior work on OCR by clustering, we have developed novel methods for probabilistic clustering. These methods promise to make OCR more robust to font variations and novel document degradation conditions, and they also have applications in other classification problems where the distribution of test samples may differ from the distribution of training samples. We describe some experiments demonstrating that the approach outperforms traditional classification methods in an OCR task.

4.1 Document Image Decoding with Language Models

Until the last few years, DID had no mechanism to express prior preference for linguistically valid

strings as recognized output over invalid strings. We have considered several approaches, settling on a class of approaches in which soft linguistic constraints are expressed by a sequentially predictive probability distribution over characters, conditioned on a fixed number of previous characters (typically four). This probability distribution is called a *language model*. Paths now have their edges scored with both a match component and a language model component. In principle, the trellis must be vastly expanded so that nodes can now encapsulate linguistic context in addition to position in the image. One approach is to think of the expanded trellis as a full tree, and apply an approximate search procedure to find a nearly-best path. The approximation comes about because of the practical necessity of avoiding searching the full tree of all possible messages. We have examined one such technique, the Stack algorithm, which is widely used in speech recognition [8] and in convolutional decoding [9], and found it to be promising [20].

We now discuss our simulation of the Stack algorithm. For each level σ of noise considered, the expected slope $H(\sigma)$ of the path score as a function of pixel position is estimated as the sum of a per-pixel-column language model component and a per-pixel-column likelihood component. The former is based on an empirical entropy rate obtained by scoring the training data with the language model and accounting for the variable widths of the characters. The latter is calculated on the basis of the noise process and the marginal distribution of uncorrupted pixel values. Figure 3 indicates the sensitivity of the space-complexity of the algorithm to the choice of tilt function slope, as well as a large difference in space-complexity for two different lines at a fixed slope value, suggesting a generally large variability of space-complexity from line to line for a given slope value.

In the experiments that compare the Stack algorithm to ICP and Viterbi, the tilt function is given in terms of the expected slope $H(\sigma)$. Specifically, the slope is set at $\alpha H(\sigma)$, where α ranges over the values 1.0, 1.5, and 2.0. The Stack algorithm using a particular value of α is labeled *Stack- α* in the graphs. To prevent a computational explosion for the occasional text lines for which the specified slope turns out to be too high, a strict upper limit of 10^6 nodes is imposed on the size of the graph for any one text line. When that limit is exceeded, then the slope is reduced in steps of $H(\sigma)/10$ until the limit is obeyed.

The basic ICP algorithm has no parameters besides the language model itself, and its implementation directly follows the description given in Section ???. We have equipped the algorithm with a mechanism for early stopping so that the best path

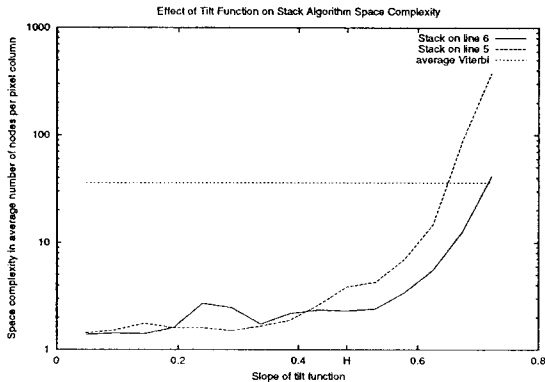


Figure 3: Space-complexity of the Stack algorithm measured in graph nodes per horizontal pixel position, as a function of tilt slope, for two different text lines both corrupted with the same noise level $\sigma = 75$. The “ideal” slope predicted on the basis of empirical entropy rate is indicated by H. Also shown, for reference, is the space-complexity of the standard Viterbi algorithm on the basic DID trellis (i.e., without a language model).

found prior to termination can be recorded and returned, but we have not had occasion to use early stopping in the experiments reported on here. It is mentioned because this capability can be important in practice.

Figures 4-7 characterize the performance of the various algorithms as a function of noise level on the first 50 lines of the test data, with the exception that for ICP at $\sigma = 100$, only the first 32 lines are used. Figure 4 presents error rates, calculated as the minimum average per-character number of substitutions, insertions, and deletions, each weighted equally, that must be performed to transform the recognized text into the original. The error-rate performance of ICP stands out; it is consistently and significantly better than the others.

The corresponding space-complexity of each algorithm as a function of noise level is illustrated in Figure 5. The extremely low complexity of Stack-1.0 makes it attractive in very low-noise situations, while higher noise levels are likely to require an adaptive tilt-slope selection procedure to achieve good performance with low complexity. Note that the ICP algorithm grows the trellis very little beyond what is already required for Viterbi; the excess is noticeable only at the highest noise levels. The time-complexity of ICP versus Viterbi is harder to gauge. ICP requires evaluating at least N complete paths versus only one for Viterbi. This number grows with noise level as shown in Figure 6, gradually for typical noise levels and more quickly as the level of cor-

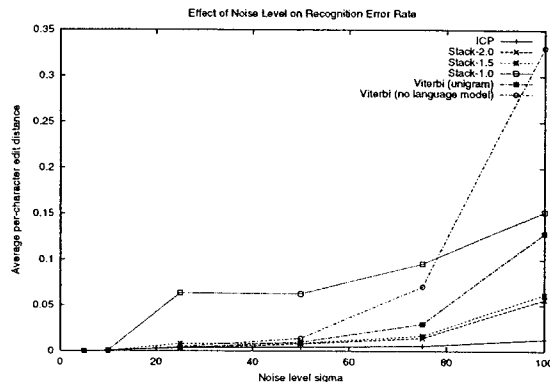


Figure 4: Error rates for the various algorithms, measured in edit distance normalized by the length of the text, using unit costs for insertions, deletions, and substitutions.

ruption becomes extreme. However, the additional path evaluations required by ICP reuse the likelihood component of the edge scores, which can be memoized. On the other hand, the language model components of the scores need to be computed for each pass, and the cost of doing so depends on the details of the language model. A careful analysis of ICP time-complexity would involve the details of how the language model is evaluated, particularly the bound functions; for now we simply observe that the time-complexity of each additional scoring pass in ICP is variable.

For both the Viterbi and Stack algorithms, time-complexity is directly proportional to space-complexity. Relative to the Viterbi algorithm, the Stack algorithm saves a factor of $|A|$ per node in time-complexity if we assume the cost of scoring an edge is the same in both cases, because in Stack the outgoing edges are scored only when a node is taken off the priority queue and expanded to $|A|$ new nodes. In practice, the cost of scoring an edge is slightly higher in Stack than in Viterbi because of the presence of a language model, and a detailed analysis of time-complexity (not attempted here) would have to take this difference into account.

Finally, Figure 7 characterizes the performance of the algorithms in terms of what they are actually trying to optimize: the average path score (shown relative to the score of ground truth, and normalized by path length). Note that ICP scores slightly better than ground truth at the highest noise level. This is because at such a high noise level, ground-truth ceases to be a highest-score path. Had a sufficiently steep tilt function been used with Stack, it too would have found a highest-score path.

We have also developed an iterative algorithm,

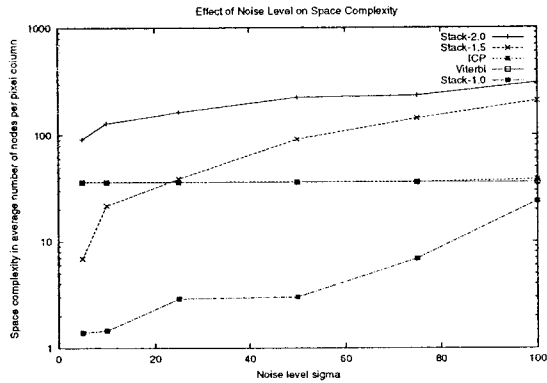


Figure 5: space-complexity as a function of noise level for the various algorithms.

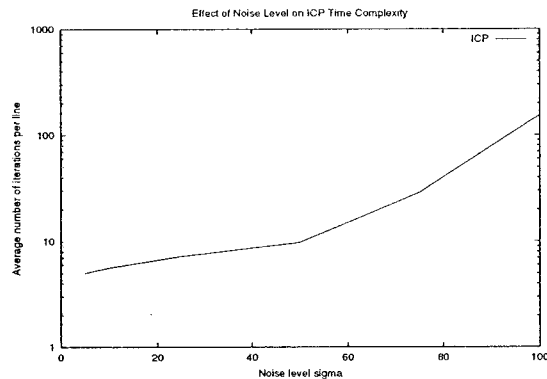


Figure 6: Number of ICP iterations required as a function of noise level.

much in the spirit of the ICP algorithm described in Section ???. We will refer to it here as a *generalized ICP algorithm*. Rather than re-score edges on each iteration, nodes are added to encapsulate additional linguistic context along paths that are deemed promising, based on lower-order upper bounds on the language model scores. As the context approaches the full context exploitable by the language model, the upper bound scores approach and ultimately reach the true language model scores. When a path is found having only true language model scores on each edge, that path can be concluded to have the highest score among all paths, and the algorithm terminates. This algorithm has the advantage over the Stack algorithm and other approximate-search algorithms that it results in a true best path, but has the disadvantage that its computational complexity is strongly data-dependent. On the other hand, it can be set up to remember the best path seen so far, and to output that upon early termina-

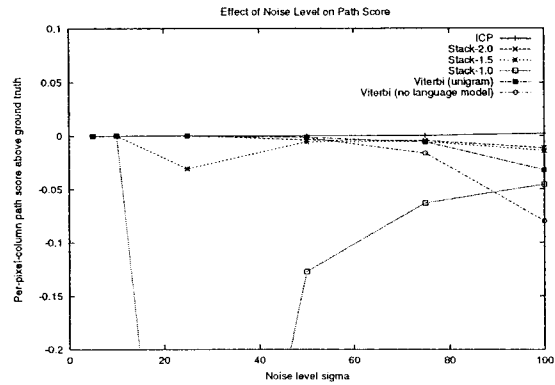


Figure 7: Average edge scores along paths found by the algorithms discussed in the text, relative to the average edge scores for ground truth.

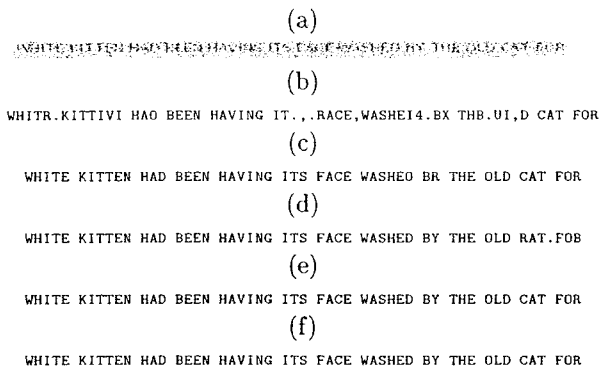


Figure 8: Example of the effect of integrating a language model into document image decoding, using several different strategies. (a) degraded, subsampled grayscale synthetic text line image; (b) decoding without a language model; (c) unigram language model via Viterbi; (d) Stack algorithm; (e) generalized ICP algorithm; (f) ground truth.

tion. In other words, to limit computational complexity, it can be set up as an *any time* algorithm for approximate best-path search.

Figure 8 shows how language modeling in its various forms can influence DID recognition accuracy. The text line used in this example was severely corrupted by additive noise to make the error rate high enough so that the differences would be clear. The text line shown in (a) is for illustration and is actually *less* noisy than the one used for recognition, which is visually unintelligible. Both the Stack algorithm and generalized ICP yield high accuracy in this example. For more details on these approaches, see references [20] and [19].

4.2 OCR By Clustering¹

This work examines the effects of clustering character images prior to recognition in optical character recognition (OCR) of printed documents. This approach has a long history in OCR, and prior work has addressed the questions of how to build a clustered representation quickly [5], as well as how to label the resulting clusters. Clustering, mixture models, and mixture-based Bayesian recognition itself, of course, has a long history in statistics and pattern recognition. In this work, we make a connection between the two approaches. The key point is that the clustering of the character templates is, in effect, a mixture density estimation of the sample distribution. This connection allows us to reexamine issues of cluster validity, style adaptation [21], and cluster label assignment within a Bayesian framework.

4.3 An Exemplary OCR Problem

For the purposes of this work, we will define the OCR problem in the following simplified manner. We assume that there is a fixed, finite set of characters (digits, lower case letters, upper case letters, special characters). Furthermore, we assume that there is an open-ended set of possible styles, where the notion of style encompasses character properties like font, size, and idiosyncracies of the particular rendering engine used. Picking a character and a style uniquely determines an idealized image (bitmap) for the character. During document creation, this idealized bitmap is printed on a piece of paper. When the document is scanned back in again, a degraded bitmap of the character, is obtained, usually by the addition of noise, blurring, thresholding, sampling error, and various forms of geometric distortions [10]. The core function of an OCR system is (roughly) to find the most likely character and style corresponding to such a degraded character image.

Traditionally, OCR systems perform this task by estimating posterior probabilities like $P(\text{char, style}|\text{bitmap})$, say, using a neural network or a Gaussian mixture model. However, estimating such probability distributions requires a large number of example characters. In practice, however, training data for many styles (fonts, degradation parameters) is not available at all.

4.3.1 OCR by Solving a Cryptogram

An alternative approach proposed in the literature [5] is based on the idea of clustering similar character shapes and then assigning character labels to the resulting clusters. Such an approach is attractive

¹This section is based on, and contains excerpts from, a paper presented at SPIE '2001[4].

because the clustering process itself is font independent, and cluster labels can, ideally, be assigned independent of the actual bitmap representation of the characters. This, on ideal data, such an approach is completely font independent and automatically generalizes to arbitrary unknown fonts. Clustering is also attractive because of the emergence of token-based compression methods that already represent documents as a collection of tokens. If we can carry out recognition directly on these clusters, we can perform OCR directly on token-compressed data.

However, in practice, such methods for carrying out OCR by clustering have not been very successful. The reason is, this paper argues, that the clustering methods used have modeled the actual statistical nature of the recognition problem poorly. This work describes how to begin combining the advantages of font independence of clustering OCR systems with the robustness of statistical methods used in current commercial OCR systems.

4.3.2 Gaussian Mixture Models

Let us assume, for the purpose of illustration, that each character image in the input document is represented by a feature vector \tilde{v} that is derived from a prototype feature vector $v_{c,s}$ representing character c and style s corrupted by an additive error G with zero mean and Gaussian distribution. In such a framework, the class conditional densities are $P(v|c,s)$ are then Gaussians. If we take a supervised pattern recognition approach, we estimate the class conditional densities (or, equivalently, priors and posteriors) from training data, derive discriminant functions, and use those to classify each unknown feature vector \tilde{v} in a Bayes-optimal sense as one of the different classes c, s .

The problem with this approach is that estimating the class conditional densities depends on a representative sample of degraded feature vectors over character classes c and styles s . If we do not have such a representative sample, our class conditional density estimates are going to be poor and recognition accuracy suffers.

We can, however, take a different approach using a partially unsupervised method involving the sample distribution. In a real OCR problem, we are usually given not a single character to classify, but many thousands of samples, one for each character in the document. Of course, these characters are not labeled, so we cannot derive the class conditional densities from this sample. However, what we can do is model the sample distribution, that is, the distribution of degraded feature vectors \tilde{v} , ignoring their class labels. If we assume that the class conditional densities are Gaussians, then the sample distribution is going to be a Gaussian mixture

4.3.3 Experiments

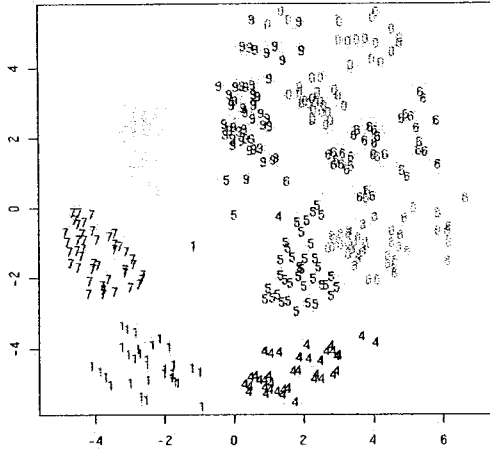


Figure 9: Low-dimensional representation of character feature vectors.

component (the frequency of the classes, c, s , are the mixture parameters). If we then try to recover the mixture components of this Gaussian mixture, we recover the individual class conditional distributions. We still do not necessarily know the classes corresponding to each such class conditional distribution, but we have a lot more information at our disposal to assign such labels than if we tried to classify characters one-by-one.

This idea is illustrated in Figure 9, which shows a two-dimensional representation (a Sammon mapping) of the feature vectors representing severely degraded samples of digits from a single font. In the figure, class assignments of the different samples are indicated by colors and labels. Looking at clusters with this information corresponds to estimating the class conditional density given labeled training data. But it is clear that even if we do not have labels available, the data still falls into fairly distinct clusters; recovering these clusters without using class labels corresponds to the clustering OCR described in this paper. (In their original, high dimensional space, these clusters are considerably better separated than in the low-dimensional non-linear mapping shown in the figure.)

If we compare this to previous methods for OCR by clustering, what it means practically is that we replace the ad-hoc, non-statistical clustering methods used in the literature [5] with a Gaussian mixture estimation algorithm. Furthermore, statistics can also help us with additional questions that the original OCR approaches left unanswered, most importantly: how many clusters should there be?

To study the feasibility of this approach to OCR, a simple prototype system based on the ideas described in this paper was implemented and applied to 1500 images of digits in the cmr6 font from the Bell Labs database of severely degraded character images found on the University of Washington Database I. The input data was divided into 1000 training samples and 500 test samples. The images were centered, convolved with a Gaussian of $\sigma = 1$ and subsampled to a size of 10×10 . The resulting image was treated as a raw feature vector and 7 principal components were extracted. These PCA feature vectors were then used as input to a sample-distribution based classifier, as well as a mixture discriminant analysis-based classifier (MDA; [7]).

The clustering OCR system performs its unsupervised clustering using the method described by Frelley and Raftery[6] and implemented by the `mclust` package for the R statistical system. Clusters in this approach are represented as Gaussian distributions. The method first performs hierarchical clustering and follows it by Expectation-Maximization (EM) steps to optimize the cluster shapes. In these experiments, the cluster shapes considered by the algorithm were “spherical” (all clusters have spherical covariance matrices), “uniform” (all clusters have the same covariance matrix), and “unconstrained”.

In this way, the clustering OCR represents the unlabelled sample distribution as a mixture of 15 Gaussians. By assumption of the method, each Gaussian corresponds to a single digit label. When the assignment of labels to clusters is correct (either based on cryptanalysis or based on a non-specific classifier), the error rate of the sample distribution based recognizer on test data is 0.7% ($N=500$) in these experiments.

To compare the performance of the clustering OCR with a traditional approach to character recognition, a Mixture Discriminant Analysis (MDA) model[7] was trained. An MDA model represents likelihood functions as mixtures of Gaussians and uses Bayes rule to perform classification. The Gaussian mixtures are estimated using the Expectation Maximization (EM) algorithm. In the experience of the author, as well as based on results reported in the literature[7], MDA performance is roughly comparable to the performance of other, commonly used classifiers like neural networks and radial basis function methods. The R implementation of MDA (available from the R web site) was used for the experiment. When the MDA classifier was trained on the training set ($N=1000$), its error rate on the test set was 0.6% ($N=500$)

4.3.4 Discussion

Ideas of clustering and style in OCR are not new and have been explored by a number of authors explicitly or implicitly. What this work contributes is a re-examination of clustering OCR methods from the point of Bayesian statistics, Gaussian mixtures, and mixture density estimation of the sample distribution. This helps both understand why and how clustering OCR methods work, and helps us improve them. The long term promise of this work is to arrive at classification methods that are considerably more robust to statistical differences between training and test data than traditional pattern recognition methods. The initial experiments presented above suggest that such an approach is feasible; more sophisticated implementations are needed to demonstrate that it delivers superior performance in real-world situations. For OCR systems in particular, this translates into much more robust recognition when novel fonts or document degradation conditions are encountered.

4.4 Classification by Probabilistic Clustering

A key limitation of trainable classifiers is that they can be sensitive to novel data whose distribution is significantly outside the training set. We have seen above how modeling the sample distribution using Gaussian mixtures can achieve comparable font independent performance to existing classification methods. Building on those ideas, we have developed a novel, non-parametric probabilistic clustering technique.

In any document each character class is usually represented by a few shapes representing a few fonts. If character images can be automatically clustered into equivalence classes, each representing a font-class pair, the OCR problem reduces to assigning class labels to each equivalence class. This can be achieved by solving a cryptogram (from linguistic clues) or by bootstrapping with some other OCR method. The advantage of clustering OCR is that it relies on font consistency alone, without modeling fonts themselves. As a result it works equally well on previously unseen fonts. Breuel [?] proposes a novel clustering algorithm where the similarity function is an estimate of the probability that two images have the same font and class labels. This probability function can be learned automatically from training data by any of several techniques, such as mixture models or multi layer perceptrons.

The approach is based on modeling, using a multilayer perceptron (MLP), the probability that two given images represent the same character. These probabilities are then integrated into an overall interpretation of a document using the maximum like-

lihood assignment of character identities to the individual images in the maximum entropy distribution compatible with the pairwise probability estimates derived from the MLP. Experiments have demonstrated superior performance on a font-independent recognition task compared to traditional pattern recognition problems.

5 Grayscale Document Image Decoding

The emergence of low-cost handheld digital cameras as a viable means of document image acquisition motivates the extension of the DID to function on relatively low-resolution grayscale images. Doing so involves significantly generalizing the channel model used by DID. One approach [18] involves carrying out the search in a high-resolution hypothesis image domain, and simulating the physical sampling and noise processes to match against the observed image. Initial results on a challenging test case are promising; the technique was found to perform favorably relative to the simple alternative of adaptive thresholding followed by application of a standard commercial OCR product. Figure ?? shows the test image used for this experiment. The edit distance between ground truth and the result of grayscale DID (with unit weighting for substitutions, insertions, and deletions) was seventy, versus ninety-one for binarization followed by commercial OCR. While preliminary, these results are felt to be encouraging.

6 Conclusions

We have summarized PARC research on document image understanding technology which has developed software tools which have achieved high accuracy on images of documents printed in a wide variety of typefaces and page layout styles, and exhibiting severely degraded image quality.

Our principal method of attack has been **retargeting**: that is, trainable to individual documents or sets of similar documents. In practice, the effort of manual DID training is low, requiring segmentation of page images into text-lines only and the presentation of training ground-truth as text-lines parallel with the images. In this paper we have also proposed **scaling up** the DID methodology using ensembles of automatically pre-trained DID decoders, to reduce training further. We have also made recent progress towards **adapting** systems, in which recognizers, without any manual training, adjust themselves to the document at hand: in this way, training may someday be reduced to zero. Finally, we have extended DID methods to handle *gray-level images*.

All of these R&D projects are ripe for extension, experimentation, and application to government-

agency needs.

References

- [1] H. Baird. Model-directed document image analysis. In *Proceedings of the DOD-sponsored Symposium on Document Image Understanding Technology (SDIUT 1999)*, pages 42–49, Annapolis, Maryland, April 1999.
- [2] D. Bloomberg, T. Minka, and K. Popat. Document image decoding using iterated complete path search with subsampled heuristic scoring. In *Proceedings of the IAPR 2001 International Conference Document Analysis and Recognition (ICDAR 2001)*, Seattle, WA, September 2001.
- [3] T. Breuel and K. Popat. Recent work in the document image decoding group at xerox parc. In *Proceedings of the DOD-sponsored Symposium on Document Image Understanding Technology (SDIUT 2001)*, Columbia, Maryland, April 2001.
- [4] T. M. Breuel. Modeling the Sample Distribution for Clustering OCR. In *SPIE Conference on Document Recognition and Retrieval VIII*, 2001.
- [5] R. Casey, S. K. Chai, and K. Y. Wong. Unsupervised construction of decision networks for pattern classification. In *Proc. ICPR-7*, July 1984.
- [6] C. Fraley and A. E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. Technical Report No. 329, Dept. of Statistics, U. of Washington, February 1998.
- [7] T. Hastie and R. Tibshirani. Discriminant analysis by gaussian mixtures. Technical report, AT&T Bell Laboratories, 1994.
- [8] Frederick Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, Massachusetts, 1997.
- [9] Rolf Johannesson and Kamil Sh. Zigangirov. *Fundamentals of Convolutional Coding*. IEEE Press, 1999.
- [10] T. Kanungo, H. Baird, and R. Haralick. Estimation and validation of document degradation models. In *Proc. 4th Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, NV, April 1995.
- [11] G. Kopec. An em algorithm for character template estimation. submitted March 1997; returned for revision, but not revised due to the author's death; available from PARC by request.
- [12] G. Kopec. Multilevel character templates for document image decoding. In L. Vincent and J. Hull, editors, *Proceedings of Document Recognition IV, SPIE vol. 3027*, 1997.
- [13] G. Kopec and P. Chou. Document image decoding using markov source models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-16:602–617, June 1994.
- [14] G. Kopec and M. Lomelin. Supervised template estimation for document image decoding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-19(12):1313–1324, December 1997.
- [15] G. Kopec, M. Said, and K. Popat. N-gram language models for document image decoding. In *IS&T/SPIE Electronic Imaging 2002 Proceedings of Document Recognition and Retrieval IV*, San Jose, California, January 2002.
- [16] M. Lomelin. *Character Template Estimation from Document Images and Their Transcriptions*. PhD thesis, MIT, Cambridge, Massachusetts, June 1995. M.S. Thesis.
- [17] Thomas P. Minka, Dan S. Bloomberg, and Kris Popat. Document image decoding using iterated complete path heuristic. In *Proceedings of IS&T/SPIE Electronic Imaging 2001: Document Recognition and Retrieval VIII*, San Jose, CA, January 2001.
- [18] Kris Popat. Decoding of text lines in grayscale document images. In *Proceedings of the 2001 International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)*, Salt Lake City, Utah, May 2001. IEEE. To appear.
- [19] Kris Popat, Dan Bloomberg, and Dan Greene. Adding linguistic constraints to document image decoding. In *Proc., 4th International Workshop on Document Analysis Systems*, Rio de Janeiro, Brazil, December 2000. International Association of Pattern Recognition.
- [20] Kris Popat, Dan Greene, Justin Romberg, and Dan S. Bloomberg. Adding linguistic constraints to document image decoding: Comparing the iterated complete path and stack algorithms. In *Proceedings of IS&T/SPIE Electronic Imaging 2001: Document Recognition and Retrieval VIII*, San Jose, CA, January 2001.
- [21] P. Sarkar. *Style Consistency in Pattern Fields*. PhD thesis, Rensselaer Polytechnic Institute, May 2000.
- [22] P. Sarkar, H. S. Baird, and X. Zhang. Training on severely degraded text-line images. [submitted to] IAPR Int'l Conf. on Document Analysis & Recognition, Edinburgh, August, 2003.

Automated Data Extraction from Structured Documents

Janusz Wnek

Science Applications International Corp.
1953 Gallows Road, Vienna, VA 22182

Abstract

Documents, such as invoices, purchase orders, insurance forms, and bills of lading are resistant to processing by traditional automated forms processing systems. This is due both to the large number of different form types present in the input, as well as to the layout of the individual forms.

Conventional forms processing products rely on strictly structured templates. They utilize manually defined form "landmarks" to locate and extract data, and require skilled operators and significant time investment to develop and verify rule assignments that recognize document, form, and data fields. In order to process multi-page documents, such documents have to be divided and processed on a single-page basis. This causes duplication in template definition effort and resources, adds complexity to the system, and introduces the possibility of creating inconsistent descriptions.

ML-FromForms endeavors to significantly enhance automated data extraction. We assumed minimal user intervention during the entire process. This was achieved through the integration of OCR, document identification and data extraction. The data extraction method (DataX) employs general templates learned by the Inductive Template Generator (InTeGen). The InTeGen method utilizes inductive learning from examples of documents with identified data elements. The system was validated on a variety of flexible forms, including insurance forms and invoices.

1 Introduction

Data extraction from documents has been an active research and development area since the introduction of OCR. Since then, many developed systems extract data with various degrees of automation. Given a data extraction system, one of the most difficult tasks in operation is the process of "setting up" the system or the process of defining data elements for extraction.

The conventional systems utilize manually defined form "landmarks" to locate and extract data, and require skilled operators and significant time investment to develop and verify rule assignments that recognize document, form, and data fields. In order to process

multi-page documents, such documents have to be divided and processed on a single-page basis. This causes duplication in template definition effort and resources, adds complexity to the system, and introduces the possibility of creating inconsistent descriptions.

Application of form processing methods to complex documents may require manual or semi-manual preparation of complicated models, which make them prohibitively expensive due to labor costs. For example, [1] describes a system for an automated evaluation of invoices. The purpose of the system is to detect and recognize price entries of item tables within invoices. The system is not model-driven but it makes many assumptions about the structure of the invoices and data elements constituting a table. It does not attempt to extract other, non-table-like data elements. Consequently, the system seems to be restricted to a very specific category of documents.

A more flexible system for processing invoices is presented in [2]. The system consists of two components, an OCR tool, and a data extraction component that contain declarative knowledge about the domain. The system uses special language for describing objects in structured documents. The drawback of this method is that document models have to be carefully crafted and described by a document engineer. In an evaluated domain of insurance invoices, the system achieved 50% automation rate with an error rate below 1% on selected data elements. Even though the system has a potential for processing diverse documents, the document definition process seems to be complex.

SmartFIX [3] is a commercial document analysis system capable of processing mixed-format documents including forms, invoices, and letters. The main installation of the system was done for processing healthcare documents in Germany. Despite its initial success the system is difficult to setup for new processing domains.

In contrast to those systems, the ML-FromForms system completely automates the process of document definition through machine learning. The method is applicable to a variety of flexible form documents, and provides easy and fast document type definition.

The form and field definition module of ML-FromForms is easily manipulated to identify and recognize the various data elements for additional processing. It automatically develops and updates form identification "rules" to recognize a form type and distinguish it from hundreds of others. By "rubber-banding" the specific data elements and identifying the fields of data to be analyzed, a user can readily input form data. Unlike other form data content identifiers, ML-FromForms can be instructed to learn the varying characteristics of data and react in a known, intelligent, and predictable fashion. Its Graphical User Interface allows rapid form characteristic definition and then provides automated assistance to learn varying form/data content and construction.

The ML-FromForms document identification/training module assumes the structure of the document domain (relationship between potential data elements in invoices, auto insurance forms, home insurance forms, etc.) and the organization of training documents according to templates (document types) as a starting point for deriving document type descriptions. A few representative training examples from each document type training set are selected, a general description of a document type is built, and the description is stored in a template definition file. The training process and derived template definition, utilize textual and positional information obtained from scanned OCR data. As a result of the training phase, two templates per document type are stored: one for document identification, and the second for data extraction from documents of that type.

2 User's Input

High level of automation is inversely proportional to user's interaction with a system. In the case of ML-FromForms user's input is simple and minimal. The user assembles sets of training documents, scans them and enters them into the system in an organized way. The documents have to be sorted according to their type and their pages ordered. The user also assigns a name to each document type (template) in the process of importing images into the system.

For the purpose of data extraction, the user identifies data elements on the screen by "rubber-banding" the specific data elements and assigning field names. No other input, such as type of the data, location, context, format, etc. is required. ML-FromForms constructs such information automatically from the document samples.

Figure 1 shows an example of data element definition in ML-FromForms. The user just selected and labeled Billing Address field. With the cursor over the field, the system displays the name and the OCR'd contents.

3 Image Processing and OCR

All documents processed by ML-FromForms undergo some elementary image processing steps, which are performed automatically. First, image orientation is

detected and the image is rotated accordingly. Second, image skew is detected and the image is de-skewed, if necessary. Third, vertical and horizontal lines are removed. Fourth, the image is de-speckled.

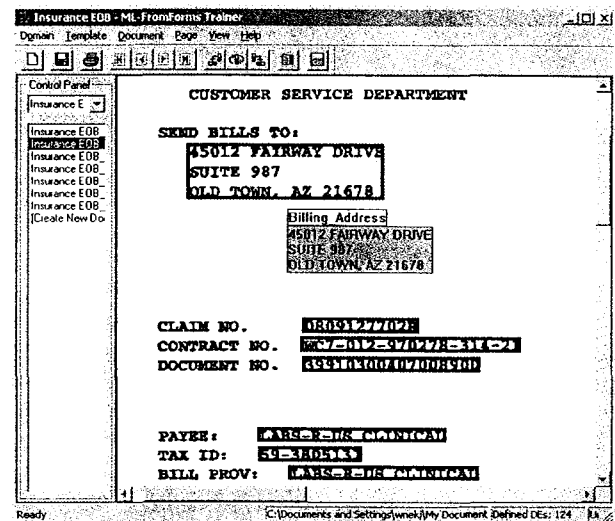


Figure 1: Data element definition in ML-FromForms.

OCR is performed on the complete document. Detailed document descriptions, segmented into pages, lines, words, and characters with accompanied positional information, are stored along with the document images. ML-FromForms uses OCR data in multiple ways. First, it is used by the graphical user interface to feedback to the user the textual contents of a selected field. This feature helps in assuring quality of the data submitted for training. Second, the inductive document classifier uses OCR data to create descriptions of document types. Third, the inductive template generator uses the OCR data to create descriptions of data elements for extraction.

4 Inductive Document Classifier

Document classification plays an important role in the data extraction system. It narrows the scope of potential actions and directs the subsequent actions by indicating the document type for data extraction.

The primary discriminatory features used by the classifier to describe document types are byproducts of optical character recognition. The classifier uses words and characters, their location and, their order to generate general descriptions. The generalization of descriptions occurs over the set of training documents collected and assembled for each document type.

Generalization of document descriptions involves intersecting two sequences of words and then for the common words, generalizing their values. For example, given the two sequences of words, T_CAI, D_CAI, below, describing the contextual attribute (CA) of data element (i) on template (T) and document (D), their

generalization results in T_CAi . Spacing between words was added to highlight the method of aligning words.

Table 1: Generalization of two sequences.

| | | | | | | | | | |
|--------|----|----|----|----|----|----|----|----|----|
| T_CAI: | w1 | | w2 | w3 | w4 | w5 | w6 | | w8 |
| D_CAI: | w1 | w8 | w2 | | w4 | | w6 | w3 | w8 |
| T_CAI: | w1 | | w2 | | w4 | | w6 | | w8 |

The method for aligning words incorporates three general preference criteria for selecting words for alignment. Additional criteria may be added based on document characteristics in a particular document domain. The general preference criteria include:

1. Words in the template are more significant than those in the document (unless the template was not generalized yet).
2. Alignment should maximize the number of common words.
3. Alignment should take into consideration keywords or key phrases delimiting identifiable sections of a document.

5 Inductive Template Generator

A template for a given document type is built in an incremental way from examples of documents (training documents) and data element (DE) definitions provided by the user. At the minimum, DE definitions consist of five elements only: vertical and horizontal position of the bounding box origin, width, height, and data element identifier.

5.1 Induction

The procedure for generalizing templates iterates through the list of data element definitions and generalizes each data element description. Generalization of DE descriptions is performed independently of each other.

Given the *first training* document and a set of data element definitions, InTeGen initializes the template by copying the document description to the template description, and by constructing and attaching initial data element descriptors to the template for each data element. Each DE descriptor is constructed based on its definition and document contents: filling out the descriptor involves extraction of relevant lexical descriptors from the document description, determining values of physical and control attributes, and constructing contextual attributes. Initially, contextual attributes comprise all document words. For each attribute, all types of distance measurements are taken, i.e. pixel distance, and sequential and vertical word distances. Copying the document description to the template is simple because both document and template descriptions share the same data structures.

Presented with the *next training* document and the set of DE definitions, generalization with the template occurs unconditionally. The subsequent training documents may or may not be generalized with the template depending on template generality, or in other words, the template's capability of correctly predicting the place and scope of its data elements. Template generalization equates to generalization of any of the DE descriptions. In order to determine if a template covers DE descriptions from the training document, a prediction test is conducted for each DE. The template's DE description is generalized only if it is not capable of correctly predicting the given DE on the training document.

DE description generalization involves generalization of particular attributes. For physical and control attributes expanding the range of measured values is sufficient. The selection of this type of generalization was dictated by the nature of document attributes, values of which are continuous rather than discrete. For example, assuming that DE sequential word distances for the first two documents were 5 and 8, the generalized distance results in a range between 5 and 8. This means that if the third document distance value is 7 then the third document is covered by that condition.

Generalization of context attributes involves intersecting two sequences of words and then, for the common words, generalizing their values. It is performed by the method for generalization of two sequences (Table 1).

After DE generalization, the DE description is evaluated with regard to its utility. There are two criteria for testing DE description utility. The first is based on description evaluation, and the second is based on performance evaluation. If both tests are passed then the DE description replaces the previous one in the generalized template. If any of the tests fail, then a new alternative description is created.

Description evaluation involves determining the amount of change in context close to the DE. The change is determined by comparing sequence word distances in the last two generalizations. If the distance to the word immediately preceding DE or the distance to the word immediately following DE increased by more than predefined thresholds, then a new alternative rule is added. The threshold values are determined empirically for the given document-processing domain.

Performance evaluation involves testing the prediction capability using the recently generalized DE description. The document that was recently used for generalization serves as testing material. This procedure allows for producing consistent and complete descriptions with regard to the training set of documents.

In some cases a data element may show a high degree of variability in addition to not having precisely defined boundaries. For example, a data element may consist of

a varied number of lines with some other data immediately following it. The attached data may easily be construed as an integral part of the DE especially when it takes place optionally. In order to detect ambiguous boundaries, InTeGen learns the structure of the edit phrases that delimit a data element.

The procedure for generalizing templates takes advantage of failing some prediction tests to learn the edit phrases. Edit phrases consist of the difference between the predicted text and the defined text. The phrases are generalized, stored and utilized when predicting data elements. The edit phrase data structure consists of a location attribute, a generalized text of the phrase, and a generalization counter. The generalized text of the edit phrase stores actual words if they reappear, or word types for varied phrase words.

5.2 Prediction

Prediction plays an important role in building a generalized template (InTeGen) and actual data extraction from new documents (DataX). Given the fact that during template generation data element definitions are available from a user as an integral part of the training set, DE definitions may also serve as a basis for comparison with predicted values. Feedback produced that way is invaluable for an automated system. It provides a means for detecting the need for performing generalization, and once generalization is done, the verification of a template's validity. As a result of such verification, it may lead to follow-up actions such as building alternative descriptions.

Given the data element identification, the data element prediction procedure selects the DE description from the template. DE description is by itself disjunctive. It consists of alternative descriptions covering variations in the DE context observed on different training documents. The procedure has to decide which alternative description is best for making prediction on the given document. In order to do this, it performs partial prediction based on all of the alternative descriptions and then selects one to complete the prediction process. In the case where no alternative description can be successfully applied, the procedure reports failure.

The prediction process breaks down into two stages: finding the origin of the data element on the current document and determining its size. Finding the DE origin is simplified to finding the first word of the data element, which is facilitated by the DE description constructed in the training phase. This task involves selecting the proper search phrase to address the first word, applying one of the distances to find candidate words, and determining the best candidate based on testing the first word properties. Once the first word is known, the origin of the DE bounding box is also known. Determining DE size is based again on

measurements and characteristics gathered during the learning phase and stored in DE description.

Finding the DE origin consists of searches for the DE using context phrases located before or after the data element. Regularly, one of the searches succeeds in finding a reliable phrase that is located in consistent proximity from the data element that yields anticipated DE origin. If the two searches do not yield an anticipated DE origin, a range of words for examination is constructed, and the DE origin is decided based on sequential location in the target document and data type of the DE first word. If there is no valid context phrase before or after the DE, or it is not possible to predict using the sequence distance, context phrases above and under are examined, and the vertical distance may be applied to predict the DE origin.

In the process of finding the DE origin, a procedure is employed to search for context phrases. Based on the template description, it constructs a multi-word context phrase that precisely addresses the DE. The phrase has to be close to the DE to assure the most precise reconstruction of DE location. In order to assure that the context phrase is not mismatched within the currently processed document, a unique landmark in close proximity to the phrase is being used. The unique landmark is a word that was assigned the best search utility during template learning. The procedure returns the address of the word nearest to the DE origin.

6 Experimentation

The ML-FromForms system is capable of processing a variety of flexible form documents. Extensive experimentation and testing for the document classifier module was presented in [4], and for the data extraction module DataX, in [5]. Here we summarize the testing methodology and results.

Figure 2 presents a sample of documents that can be processed by ML-FromForms (only first pages of multiple-page documents are shown). For some applications, templates for several hundred document types have been created and data successfully extracted.

The Inductive Document Classifier was trained and tested on a database consisting of 16,039 pages of 500 document types. The data was randomly split into two sets, each of them containing one-half from each document type. Results from two train-test sessions were combined to give a complete picture of data quality and classifier performance.

Table 2 summarizes testing results. It shows the number of correct, incorrect, and unrecognized pages (a page is considered unrecognized when none of the document types is matched above a predefined confidence level). Classification accuracy, which is the ratio of correct pages to all tested pages, is 98.3%, which is very high for the type of processed documents.

The accuracy is even higher (99.3%) when two best guesses are taken into consideration.

The main objective for the experimentation and testing in [5] was to establish prediction accuracy of the data extraction method, i.e. accuracy of finding and extracting data elements in documents similar to those presented in Figure 2. OCR accuracy was not of our concern; hence ground truth data was easy to create and consisted only of bounding boxes and identifiers of defined data elements. Another two objectives for testing were to establish the approximate number of training examples required to achieve a desired prediction accuracy, and to confirm the system's stability on known (training) data.

Table 2: Document Classifier Testing.

| Classification | Correct | Incorrect | Accuracy |
|--------------------|---------|-----------|----------|
| Best class only | 15,774 | 183 | 98.3% |
| Two best classes | 15,926 | 31 | 99.3% |
| Three best classes | 15,931 | 26 | 99.3% |

There may be many other factors that influence prediction accuracy in general. First of all, not all documents or their parts may qualify as having flexible layouts. Therefore, not all data elements may be extracted from them. Second, document image quality and OCR accuracy may influence both the quality of data element prediction accuracy, as well as data element contents.

Data extraction testing involved 10 selected document types for which ground truth data was created. For each document type up to 11 data elements were defined, totaling 103 data elements in all document types. Ground truth data was created for 20 documents in each type. In all, 2,060 data elements were defined. In each train-test session, the system was trained using from 1 to 20 training examples and evaluated using all 20 documents.

Three error types were measured: omission error, commission error, and precision error. Precision error captures imprecision in extracting the data element, i.e. detecting too small or too large a bounding box. This error type is usually of lesser concern to users than the errors of commission or omission because it may be corrected by post processing.

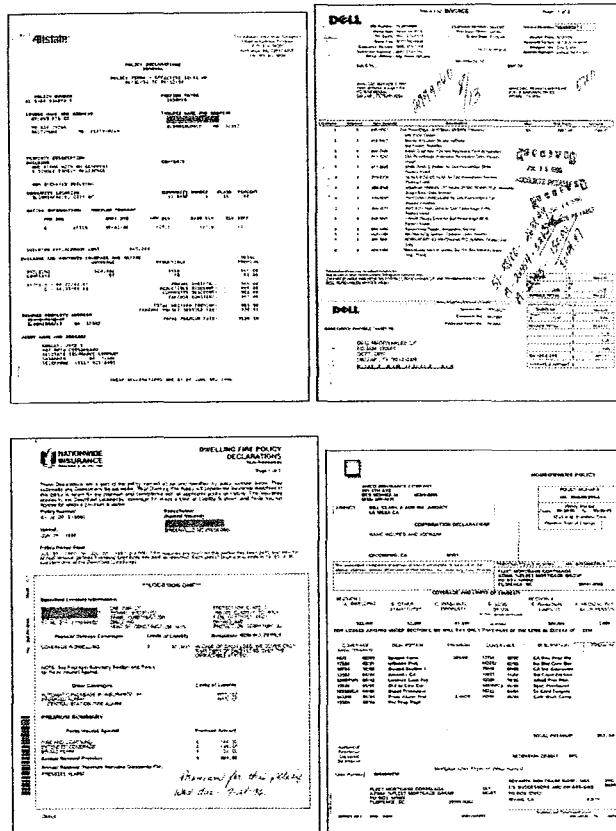


Figure 2: Examples of insurance and invoice documents.

Table 3 summarizes DataX testing for 10 document types. The overall error rate includes omission and precision error rates only, because the system did not make any errors of commission. The majority of errors are due to imprecise scope of bounding boxes. The system makes less than 5% errors (2% omission errors) after learning from about 5 training examples. (Note that at 5 training examples, the system is still evaluated on 15 unknown examples). In addition, the system achieves 0% error rate when presented with all known documents, which suggests that the template descriptions are consistent with the training data.

Table 3: Data Extraction Testing.

| Number of training examples | Omission error rate | Precision error rate | Error rate |
|-----------------------------|---------------------|----------------------|------------|
| 1 | 13% | 11% | 24% |
| 3 | 4% | 7% | 11% |
| 5 | 2% | 2% | 4% |
| 10 | 2% | 1% | 3% |
| 20 | 0% | 0% | 0% |

7 Conclusions

Data extraction from flexible-form type documents is a difficult problem because of the indefinite structure of such documents. Data fields do not have fixed locations and characteristics on a page, and may flow between pages of documents. Currently, commercially available forms processing systems still require significant user involvement in the process of defining fields for extraction. The process requires competence, continued maintenance, and is time consuming. The user has to: (1) learn how to operate a new system, (2) discover patterns in data format, and (3) learn ways of transferring this knowledge to the document processing system. The third requirement may be extremely time consuming and often frustrating given the system constraints. Moreover, the user's input, calculations and estimations may often be imprecise and lead to system errors.

The ML-FromForms system offers several advantages over other systems. The most important feature manifests itself in the minimal user's involvement in the process of defining fields for extraction. The user is only required to identify what the location of the object of extraction is. The user is not required to provide any additional characterization of the object. All necessary information needed to characterize the object and its context is acquired automatically. This minimizes human effort and chances for introducing errors. Additionally, the system supports the data element definition process by predicting DEs based on currently defined templates. This feature greatly speeds up template development, and assures template quality.

The next advantage results from robust generalization of training examples. The example documents are real documents as opposed to synthetic document models. This way the system encounters not only variability in data arrangement but also possible noise. By processing and filtering out noisy descriptions, the system becomes resilient to noise when processing subsequent documents. In addition, machine learning makes it feasible to include the template generation module in a real-time system, and acquire knowledge about new document cases during production. This creates an opportunity for the document processing system to become instantly responsive to incoming variations in document designs.

Finally, the symbolic descriptive patterns are language independent. The method could be integrated with versatile OCR devices that process documents in different languages, e.g. Latin, Cyrillic, Greek, Arabic, Japanese, or Chinese.

References

- [1] M. Koppen, D. Waldostl, and B. Nickolay, A System for the Evaluation of Invoices, in *Document Analysis Systems II*, (World Scientific, 1998) 223-241.
- [2] T. Bayer, and H. Mogg-Schneider, A Generic System for Processing Invoices, in *Proc. Int. Conf. on Doc. Analysis and Recognition*, (IEEE Computer Society Press, 1997) 740-744.
- [3] A.R. Dengel and B. Klein. SmartFIX: A Requirements-Driven System for Document Analysis and Understanding, in *Proc. 5th International Workshop on Document Analysis Systems V*, D. Lopresti, J. Hu and R. Kashi eds. (Springer, 2002) 433-444.
- [4] J. Wnek. Learning to Identify Hundreds of Flex-form Documents," in *Proc. of SPIE, Document Recognition and Retrieval VI*, D. Lopresti and J. Zhou (Eds.), Vol. 3651 (1999) 173-182.
- [5] J. Wnek. Machine Learning of Generalized Document Templates for Data Extraction, in *Proc. 5th International Workshop on Document Analysis Systems V*, D. Lopresti, J. Hu and R. Kashi eds. (Springer, 2002) 457-468.

Automated Logo Detection and Recognition

*Tom Drayer and Ken Cantwell
Department of Defense*

Multilingual Documents

Farsi Searching and Display Technologies

**Kazem Taghva, Ron Young, Jeffrey Coombs
Russell Beckley, Mohammad Sadeh, Ray Pereda**
Information Science Research Institute
University of Nevada, Las Vegas
Las Vegas, NV 89154-4021

Abstract

In this paper, we report on our ongoing research for the development of a Unicode-based search engine for Farsi. The activities consist of an I/O subsystem, Farsi stemmer, test collection preparation, and the search engine itself.

This engine is intended to be independent of the operating system platform using no special hardware or software. We are further planning to tune the system for other languages with Arabic related scripts.

1 Introduction

Farsi is the official language of Iran and one of the two official languages in Afghanistan (Pashto is the other language). In addition, Farsi is spoken in Tajikistan and parts of Uzbekistan [8]. There are more than two million Iranians living in the United States, and the Farsi language is taught at many universities and institutions in North America and Europe [3]. In recent years, many web sites have appeared that provide information in Farsi. In particular, most of Iranian newspapers and magazines have official websites with daily articles in Farsi. As more Farsi materials become available on the web, it is evident that search tools need to be developed to deal with Farsi information access needs.

The recent methodologies in Cross Language Retrieval and Machine Translation are one approach to addressing the non-English information access needs. Farsi is typically omitted from most of these methodologies due to lack of standards for Farsi fonts and input/display technology. For example, most of the Farsi websites render their information in specialized fonts or images of text. The lack of standards and the need to use the Internet without using an Arabic-based alphabet has even convinced a community of Iranians to develop a Latin-based script known as EuroFarsi [4].

In the early days of our project, we noticed that there was not much information that could help us develop our search engine. For example, there was no known Farsi stemmer, Farsi Collection, or Farsi stopword list. Also, we needed an input/display method that would allow us to enter Farsi query words in a Latin-based operating sys-

tem without any special software or hardware. It was further necessary to have a standard character encoding for text representation and searching. Most of our design decisions in this project were influenced by the above-mentioned problems.

This paper is a description of our ongoing activities in the development of a Farsi Retrieval Engine. Section 2 describes an input/output system for Farsi. Section 3 explains the development of the Farsi keyboard applet. Section 4 shows the architecture of our system. Section 5 gives a short overview of our stemmer and stopword list. Our text collection is presented in section 6. Section 7 describes the search engine based on both traditional methods and statistical language modeling [12, 20, 15]. Section 8 is the conclusion and future work.

2 Collection Level I/O

When we started compiling the test collection for our Farsi project, one problem became immediately obvious: "the anarchy of fonts." In processing the collection, we encountered documents encoded with the following:

Standard encodings:

- Mac OS Farsi
- CP 1256 (Microsoft Windows)
- ISO 8859-6.16
- UCS/UTF8 (Unicode)
- ISIRI 3342 & 2901 ([6])

Proprietary encodings:

- e.g. *Persian Nimrooz*

Other encodings:

- Adobe Distiller Embedded Fonts
- Images of text

By far, the most common encoding represented in our collection was *Persian Nimrooz* [5]. A conversion table between this encoding and UCS2 (Unicode) was developed manually. The standard encodings also have Unicode conversion tables widely available. Because of the relatively low occurrences of the other encodings, docu-

ments with these encodings were removed from the collection.

Processing the collection consists of interpreting the raw HTML document and extracting the text along with any corresponding '' tags. Once the text is extracted and converted to Unicode, it is then written as a UCS2 binary file.

A key requirement for our Farsi project was to allow the input and display of native Farsi content without requiring any special hardware or additional OS software capabilities. To satisfy this requirement the system provides the following capabilities:

- a web-browser based keyboard applet for input
- if the web-browser has the ability to process and display Unicode content, it will be used
- if the browser cannot display Unicode content, an auxiliary process will be invoked to render the Unicode content into a portable bitmap image with associated HTML to display the image in the browser.

Another area of difficulty that we encountered is that the presence of whitespace used to separate words in the document is dependent on the display geometry of the glyphs. Since Farsi is written using the Arabic script (a cursive form), each character can have up to four different display glyphs. These glyphs represent the four different presentation forms:

- isolated*: the standalone character
- initial*: the character at the beginning of a word
- medial*: the character in the middle of a word
- final*: the character at the end of a word

We found that depending on the amount of trailing white space following a final form glyph, a space character may or may not be found in the text. This situation came to light when our subject matter experts were developing our test queries. We found that since the glyphs used to display the final form of characters had very little trailing space, they were manually adding space characters to improve the look of the displayed queries.

3 Keyboard Applet

The keyboard applet is written in javascript. We also developed a java version but decided on javascript to minimize the support issues related to the Arabic script capabilities of the java virtual machines included in the various browsers. The applet displays a Farsi keyboard image with the ability to enter characters from both the keyboard and mouse. The applet also handles character display conversion and joining of the input data.

The keyboard layout is based on the ISIRI 2901:1994 standard layout as documented in an email by Pournader [14]. Figure 1 shows the keyboard applet being used to define test queries.

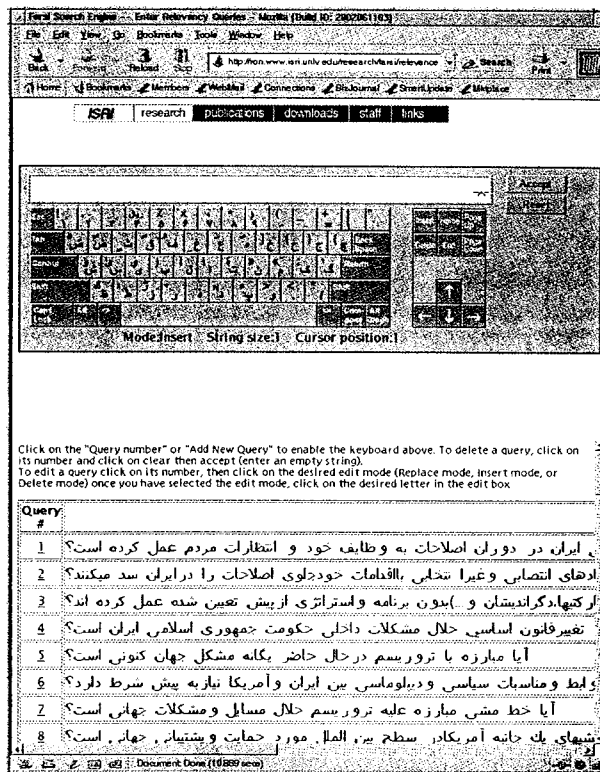


Figure 1: Example use of the keyboard applet

Display of the input data is normally performed by using the preloaded glyph images. However, if a character has not been preloaded, it can be generated on the fly. Most of the time, these generated characters are "compound" characters. Farsi (and other Arabic script languages) may use "compound" characters which are a combination of two or more separate characters. For example, the rightmost character of خُرما, the Farsi word for "date" (that is, the fruit), is a combination of a khe with a damma.

4 General System Architecture

As part of this project, we wanted to have a robust and easily extendible environment for conducting information retrieval experiments. To this end, we have developed a modular framework which allows the ad hoc loading of extended commands into a core command line interpreter.

The core command line interface provides commands to load/unload extended command libraries, user defined symbol management, and an embedded HTTP web server. The symbol table management routines are based on the ones in the PCCTS 1.3mr33 toolkit written by Parr and maintained by Moog [11]. The embedded HTTP web server that we use is libHTTPD by Hughes Technologies [16]. The cli command line interface processing routines uses the cmd command parser

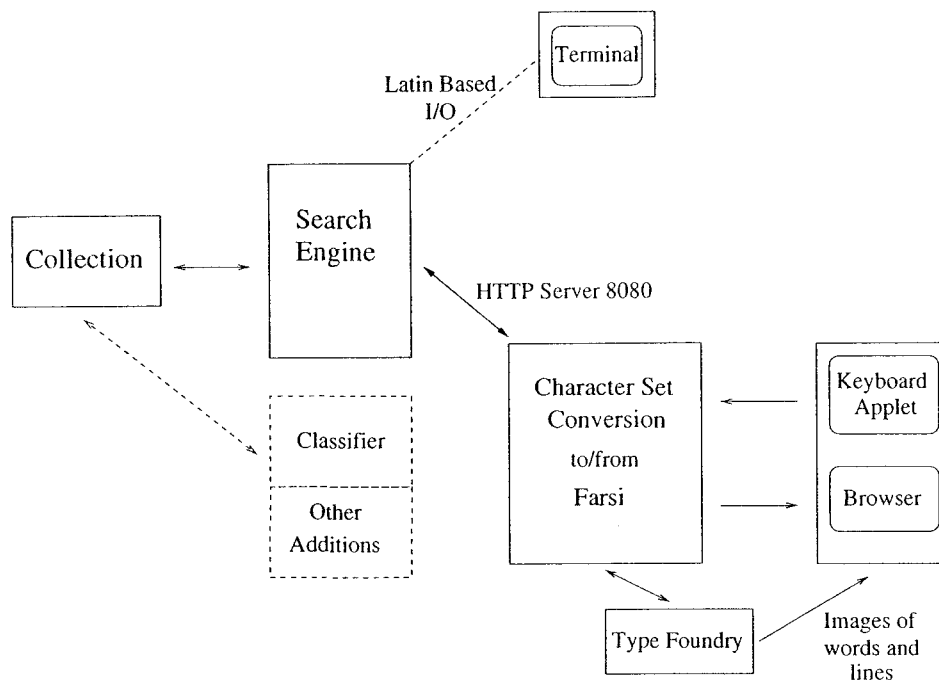


Figure 2: Farsi Search Engine Architecture

library by Columbia University [18].

By using an embedded HTTP server, we can handle non-Latin scripts easily. The search engine performs all processing with Unicode encoded documents. When an item is displayed, the Unicode encoding is sent to the embedded HTTP server. If the browser supports Unicode encoded fonts, the item is displayed using the browser. If the browser doesn't support Unicode fonts, the item is converted by the "type foundry" into a series of portable network graphics (PNG) images. We use the *freetype* engine by the Freetype Project to render the Unicode script into the images [17]. These images are then sent to the browser for display. Figure 2 depicts an overview of the system.

5 Stemmer and Stopword List

The Farsi stemmer identifies common Farsi affixes in Farsi words. Conditionally, it removes the affixes, producing an effective stem.

The Farsi stemmer is similar to the Porter stemmer [13]. For example, both are based on the morphological rules of their respective languages. Both stemmers match words with a set of suffixes and use multiple phases to conform to the rules of suffix stacking. Also, they both enforce lower bounds on how much information a stem must retain. However, there are differences. For example, the Porter stemmer counts substrings of consecutive consonants and vowels, estimating the information content, before deciding to remove a suffix. In Farsi, many spoken vowels are not written, so the stemmer cannot count them. Therefore, the Farsi stemmer uses stem length to define a lower

bound on information content (in the current version, minimum stem length = 3). Another difference is that the Farsi stemmer identifies prefixes, unlike the Porter Stemmer.

The stemmer is aggressive, removing every terminal substring that matches a known suffix, sometimes removing part of the root word. However, it rarely removes enough of a root to overconflate.

The stemmer attempts to match a suffix to a word ending by feeding the word to a finite state machine. A word may be fed the machine as many as three times, identifying as many as three suffixes stacked on the same root. The state machine reads the word backwards, i.e., it first reads the final character (the last character in reading order), then proceeds toward the first character. If the machine halts in an accepting state, we know that a suffix matches the end of the word and that the input word is long enough to retain the minimum stem length after removing that suffix. The stemmer associates each accepting state with a specific suffix and a suffix class such as *plural*.

For example, suppose the input is *نمیرفتیم* ("we were not going"). The finite state machine will identify the suffix *یم* as the longest matching suffix. The final state of the machine tells the stemmer that the suffix belongs to the *verb* class.

The suffix classes determine the modifications the stemmer makes and the additional affixes the stemmer attempts to match. The general rules for modifying words according to suffix class are:

Verb Remove the identified suffix. In accordance with the rules of prefix stacking, remove as many prefixes

| Farsi Word | English Equivalent |
|------------|--------------------|
| آن | that |
| چرا | why |
| آنجا | there |
| است | is |
| باید | must |
| بود | be |
| با | with/by |

Figure 3: Potential Farsi Stopwords

as possible while retaining sufficient stem length. The expression {ناب}{می} describes the language of relevant prefixes.

Possessive Remove the identified suffix. Feed the remainder to the finite state machine to check for non-possessive noun suffixes.

Plural Remove the identified suffix. Feed the remainder to the finite state machine to check for non-possessive, non-plural noun suffixes.

All other suffix types Remove the identified suffix.

Consider the previous example, نمیرفتیم. After identifying the suffix and suffix class, the stemmer will remove the suffix یم, leaving نمیرفت. Then it will look for the prefix ب, and not find it. It will then find the prefix ن. It will check the word length, and remove the prefix, leaving میرفت. It will then find the prefix می, check length, and remove the prefix. It outputs رفت, which is the infinitive “to go”, with the terminal ن removed.

Now, suppose the input is کتابهام, “my books.” The finite state machine will identify the suffix م, and remove it, leaving کتابها. Because م is a possessive suffix, the stemmer feeds the remainder to the finite state machine to look for more noun suffixes of the same word. Now, the finite state machine will identify the suffix ها, and remove it, leaving کتاب. Because ها is a plural suffix, the stemmer feeds the remainder to the finite state machine to look for more noun suffixes. This time, the finite state machine halts in a rejecting state, signifying an absence of more suffixes, so the stemmer returns کتاب, “book.”

In addition, we are using our Farsi document collection to generate a list of stopwords. Stopwords are identified by frequency analysis and expert judgment. Figure 3 shows potential stopwords from the list.

6 Document Collection

Our document collection consists of 1850 documents and 60 queries. These documents were collected within a six month period from many websites. Some of these websites originate in Iran and they typically represent electronic versions of popular Iranian newspapers and magazines. The rest of the articles are from websites originating in the U.S. or in Europe. These documents mainly cover topics such as politics, economic issues associated

| |
|--|
| آیا در اثر عدم رعایت ضوابط بهداشتی در ایران سلامت مردم درخطر است؟ |
| Does the lack of health standards in Iran endanger the population? |

Figure 4: An Example Query

with oil, social problems, and historical changes in the Middle East.

The sixty queries were designed by native speakers of Farsi familiar with these documents. An example query and its translation appears in Figure 4. These queries were input using our keyboard applet for the purpose of relevancy judgment gathering.

We also built a web-based application to display the queries and the articles side-by-side for our Farsi experts. After an expert reads the article, he makes binary decisions as to the relevance of the document to each query. No relevancy ranking is assessed.

7 Search Engine

Our search engine will contain several methodologies because we plan to study their effectiveness in retrieving Farsi language texts. Initially two methods will be applied: the vector space and the probabilistic.

In the well-known vector space model, documents and queries are represented as n -dimensional vectors of term weights. For a given query vector Q documents are ranked in terms of their similarity to the query as determined by the cosine measure:

$$\text{cosine}(Q, D_d) = \frac{Q \cdot D_d}{|Q| |D_d|} = \frac{\sum_{t=1}^n w_{q,t} \times w_{d,t}}{W_q W_d}$$

D_d represents the vector of a document d . The inner product $Q \cdot D_d$ is interpreted in the standard way as the sum of the products of the weights of corresponding terms in the query and document, $w_{q,t}$ and $w_{d,t}$ respectively. Weights are determined using the usual $tf \times idf$ scheme. The values W_q and W_d are the Euclidean lengths of the query and a document [19].

Our Farsi retrieval engine will also use probabilistic techniques. In a probabilistic search engine, documents are ranked by the probability that given a query q , document d should be retrieved. Using Bayes theorem and dropping the constant probability for the fixed query from the denominator, we have:

$$p(d | q) \propto p(d)p(q | d) \quad (1)$$

Recent studies have shown that the specific probabilistic approach called *statistical language modeling* has proven more effective than others on English collections [12, 20]. Some success has been shown for Arabic texts as well [9]. Statistical language modeling rejects the assumption of other probabilistic approaches that there is an underlying distribution model which the data should fit. For example, Bookstein and Swanson assumed that terms would

occur in documents with a Poisson distribution [1]. Instead, statistical language modeling uses non-parametric techniques to estimate $p(q|d)$.

Based on the concept of a language model developed for speech recognition, a query is viewed as a randomly generated set of terms. The expression $p(q|d)$ is then understood as the probability that certain terms will occur in a query given a particular model of the documents [12, 20]. Of course, queries are not typically generated randomly, but from the “perspective” of the document model, how the query is created is unknown and thus “appears” to be random.

Since $p(d)$ is typically assumed to be uniform in equation (1), the expression $p(q|d)$ alone requires an estimate. The basic approach in statistical language modeling is to begin with the maximum likelihood estimate,

$$p_{mle}(t|d) = \frac{tf_{t,d}}{DL_d}$$

where $tf_{t,d}$ is the number of occurrences of a term t in a document d , commonly called *term frequency*, and DL_d is the document length of a document d , defined as the total number of term occurrences in d , that is, $\sum_t tf_{t,d}$. However, the maximum likelihood estimate has two primary shortcomings in this context. First, the estimate will assign a probability of zero to a document if one or more query terms are not present, and second, documents only contain sparse data [12, 20, 15].

To solve these problems, *smoothing techniques* are employed. The general form of a smoothing technique can be expressed as

$$p(t|d) = \begin{cases} p_s(t|d) & \text{if } t \text{ occurs in } d \\ \alpha_d p(t|C) & \text{otherwise} \end{cases}$$

where $p_s(t|d)$ is the smoothed version of the maximum likelihood estimate, $p(t|C)$ is the “collection language model” and coefficient α_d determines the amount of probability transferred to terms not in the query. A number of techniques have been proposed to define these expressions [2, 20, 21, 12]. We plan to investigate several of these to see how they impact effectiveness of retrieval on Farsi text, especially those which move beyond the independence assumption of the unigram model [15, 10].

8 Conclusion and Future Work

The explosion in availability of Farsi websites and the eventual electronic conversion of Farsi literature are the driving forces behind our project. We believe standard fonts, operating system independence, and input/display technology are critical factors in future development of search tools for Farsi particularly. Because of the close relationship between Arabic and Farsi, our research can indirectly benefit Arabic as well. For example, over 40% of all words in Farsi are Arabic, and thus one should expect common research issues between our stemmer and an Arabic stemmer [7].

Future work includes performance testing of our stemmer and search engine. We are also planning to make detailed reports available on our input/display technology, stemmer, stopword list, and search engine implementation at www.isri.unlv.edu.

References

- [1] Abraham Bookstein and Don Swanson. Probabilistic models for automatic indexing. *Journal of the American Society for Information Science*, pages 312–318, 1974.
- [2] Stanley Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard Univ., 1998.
- [3] Center for Advanced Research on Language Acquisition (CARLA). Less commonly taught languages (LCTL) project. <http://carla.acad.umn.edu/LCTL/>.
- [4] Manou & Associates Inc. Eurofarsi experiment. <http://www.iranonline.com/eurofarsi/experiment.html>.
- [5] Neda Rayaneh Institute. Persian nimrooz. <http://neda.net.ir/downloads/fonts.shtml>.
- [6] ISIRI. Institute of standards and industrial research of Iran. <http://www.isiri.org/>.
- [7] Shereen Khoja. Stemming. <http://www.comp.lancs.ac.uk/computing/users/khoja/index.htm#stemming>.
- [8] Tore Kjeilen, Sidahmed Abubakr, and D. Josiya Negahban. Encyclopedia of the orient. <http://i-cias.com/e.o/>.
- [9] Leah Larkey and Margaret Connell. Arabic information retrieval at umass in trec-10. In *The Tenth Text Retrieval Conference, TREC 2001 NIST Special Publication 500-250*, pages 562–570, 2002.
- [10] D. H. Miller, T. Leek, and R Schwartz. A hidden markov model information retrieval system. In *SIGIR'99*, pages 214–221.
- [11] Terence J. Parr and Thomas Moog. Pcts 1.3mr33 toolkit. <http://www.polhode.com/pccts.html>.
- [12] Jay Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *SIGIR'98*, pages 275–281, 1998.
- [13] M.F. Porter. An algorithm for suffix stripping. *Program*, 14, 1980.

- [14] Roozbeh Pournader. Farsi keyboard. <http://lists.sharif.edu/pipermail/persiancomputing/2000-August/000129.html>.
- [15] Fei Song and W. Bruce Croft. A general language model for information retrieval. In *SIGIR'99*, pages 279–280, 1999.
- [16] Hughes Technologies. libhttpd. <http://www.Hughes.com.au>.
- [17] David Turner, Robert Wilhelm, and Werner Lemberg. Freetype project. <http://freetype.sourceforge.net>.
- [18] Columbia University. Cli processing routines. <ftp://ftp.cc.columbia.edu/mm/mm-ccmd-0.91.tar>.
- [19] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and indexing documents and images*. Morgan Kaufmann, 2nd edition, 1999.
- [20] ChengXian Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR'01*, pages 334–342, 2001.
- [21] ChengXiang Zhai and John Lafferty. Two-stage language models for information retrieval. In *SIGIR'02*, pages 49–56.

Porting the BBN BYBLOS OCR System to New Languages

Prem Natarajan, Michael Decerbo, Tom Keller, Rich Schwartz, John Makhoul

{prem, mdecerbo, tkeller, schwartz, makhoul}@bbn.com

BBN Technologies, Cambridge, MA 02138

Abstract

BBN's Byblos OCR system uses a language-independent Hidden Markov Model (HMM) based approach for recognizing machine printed text. A major advantage of the HMM-based approach is that it provides the ability to train models for any new language, script, or data source. All that is required for developing recognition capability for a new language is an adequate set of training and test data from that particular language. Recognition accuracy on data from different types, sources, and domains can be improved by using an appropriately selected training data set to train the system. The ability to train the system is also useful in dealing with degraded data where other, non-trainable, approaches may fail to deliver acceptable performance. In this paper, we describe our approach for porting our system to new languages and new data sources and we give some experimental results.

1 Introduction

Hidden Markov Model (HMM) based systems offer a flexible and customizable methodology for performing recognition tasks. The BBN Byblos OCR system, which uses the HMM-based BBN Byblos speech recognition engine, has been presented earlier in [1, 2]. The Byblos OCR system uses a glyph model trained on a corpus of text images, a lexicon, and a statistical language model. The ability to train the models on new types of data offers a major advantage in the context of noisy or degraded data as well as data from non-traditional sources such as text embedded in video images. Another significant advantage of HMM-based systems is that they provide a language/script-independent framework for training and recognition.

The paper is organized as follows. In section 2 we provide an overview of the process of porting the Byblos OCR system to a new language. In section 3 we describe in detail the methodology for collecting data. Section 4 provides a description of the training procedure and, in Section 5, we describe the process of system tuning (also known as optimization). Finally, we conclude with a summary of our experimental results in Section 6.

2 Porting of Byblos OCR System

The process of porting the Byblos OCR system to a new language is graphically illustrated in Figure 1. As indicated in the figure, the first step in developing support for a new language (or for customizing the recognition models to a particular data source/type for a currently supported language) is to acquire a set of scanned text images as well as a large corpus of text data in the language of interest. In the case of a new language or script, we also need to identify the alphabet set and to define a transliteration format for the text transcriptions. However, if the goal is to customize the models for an already supported language to a particular data source (for example, fax documents), then we can use the existing alphabet set and transliteration format.

The scanned text images, along with human-generated transcriptions, are used for training the glyph HMMs, while the text data corpus is used to train the language model. In generating the transcriptions for the scanned text images, it is not necessary to align the transcriptions at the character or word levels; simple line-wise alignment of the transcriptions with the images is sufficient. Word and character alignments are automatically generated by the training programs.

Training is performed using the Baum-Welch or Forward-Backward algorithm [3-5], which aligns the feature vectors with the character models to obtain maximum likelihood estimates of HMM parameters. During recognition we search for the sequence of characters that is most likely, given the feature-vector sequence and the trained character models, in accordance with the constraints imposed by a lexicon and/or a statistical grammar. The use of a lexicon during recognition is optional but its use generally results in a lower Character Error Rate (CER). The lexicon and the grammar (language model), which provides the probability of any character or word sequence, are estimated from a suitably large text corpus.

In the next section we describe the data collection procedure.

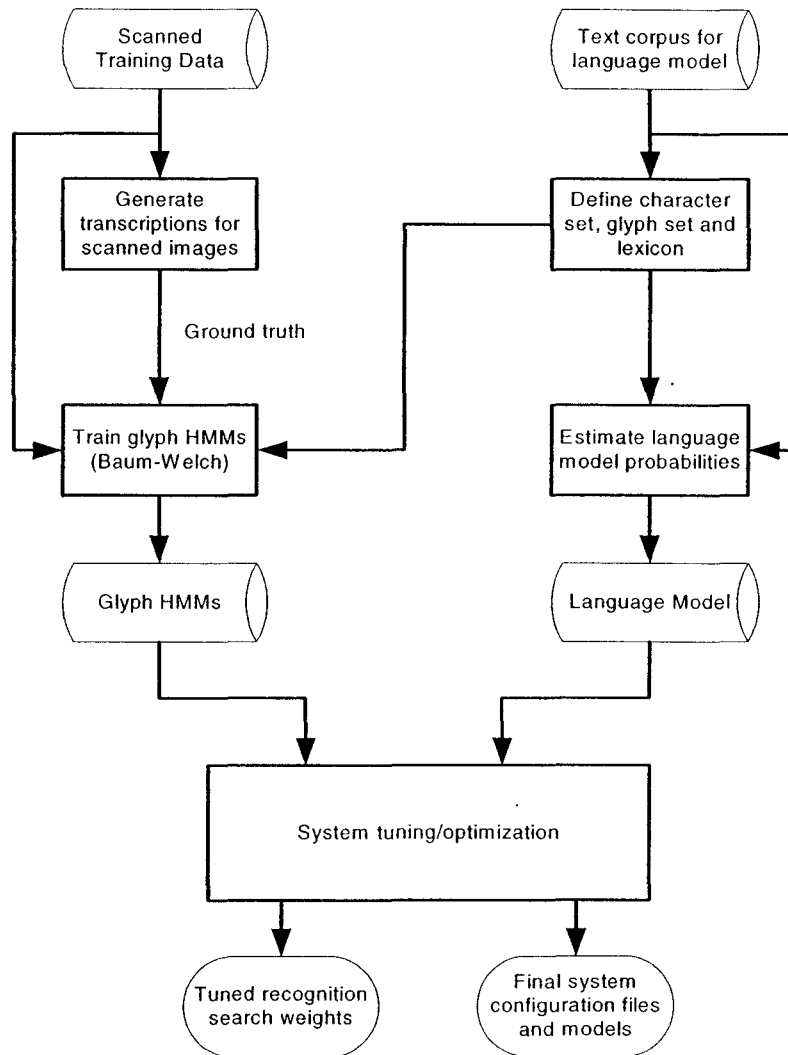
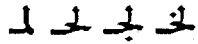


Figure 1: Porting the Byblos OCR system to a new language.

3 Data Collection

The first task in porting Byblos to a new language is to define the character set and glyph set for that language. Defining a character set for most Roman-script languages is largely straightforward because of widely accepted script standards, but languages such as Arabic, Farsi, and Pashto, which use connected scripts, present a more challenging problem because of a much more varied writing system. In all three languages, characters have context-dependent glyphs, i.e., the same character can take on different graphical forms depending on whether it stands by itself or is connected from the right, left, or both sides. Also, documents in these languages exhibit a liberal use of ligatures, where two or more characters may be represented by one glyph. It is important to take all these aspects of a particular language into consideration before defining the character and glyph sets for a language.

Unicode tables are an excellent reference for identifying a canonical character set for a language. It

has to be noted though that in many cases it is impractical to match Unicode's coverage (usually exhaustive) of a given script. For example, text in Arabic-script languages, such as Arabic, Farsi, and Pashto, may potentially include a large number of ligatures, and Asian languages can draw upon a large number of ideographs. Attempting to provide support for all possible ligatures or ideographs might be inefficient: adequate training data that incorporates them might be hard to find, and as a corollary, they might never appear in the documents of interest. Other, less exhaustive standards for document encoding can be a guide in paring down the Unicode inventory; in our past work on Simplified Chinese we have found it satisfactory to include only the approximately 4000 characters included in the legacy GB2312-80 standard. The legacy Arabic standard, ISO-8859-6, includes no ligatures, but we have included four of the most common ligatures  in our operational system.

As mentioned above, in some scripts such as Arabic, a single character can take on more than one graphical form based on whether it is connected to the right, to the left, both to the right and left, or not connected at all. Thus, of the 31 basic isolated characters in Arabic, 24 can connect to others from the right and the left, six connect only to the right, and one character does not connect at all. Each of the graphical forms, or glyphs, of a character is modeled separately. With this in mind, a glyph inventory, showing all the forms that each character may take, must then be created. Here again, Unicode and its ancillary documents (such as the "Arabic Shaping" tables) can be helpful [<http://www.unicode.org/Public/UNIDATA/ArabicShaping.txt>]. However, Unicode is a standard for characters, not glyphs, and therefore is neither normative nor exhaustive for glyph shapes. For instance, while the shaped forms of Arabic-language characters are included in Unicode (as "Arabic Presentation Forms"), shaped Pashto characters are not included. In such cases, the best reference is a font designed for the language, which typically encodes these glyphs in the Unicode private-use area.

Once the complete glyph set has been defined, a transliteration from the glyph set to 7-bit ASCII must be chosen, because Byblos uses sequences of 7-bit ASCII characters internally to represent glyphs. A phonetic transliteration can be chosen, for the convenience of developers and annotators glancing at the data (for instance, we represent the Pashto letters TAH, TEH, and TTEH as t, T, and LT, plus a numerical suffix indicating their contextual shape). For other scripts with a larger glyph set, an ASCII rendering of the glyph's index in some standard table (such as "u+0637") may be more convenient to implement. Table 1 below shows the various representations (glyph, Byblos transliteration format, Unicode) for a small sample set of Pashto characters.

Once the glyph set for a particular language is defined, the transcribers can start developing transcriptions for the training and test images. Of course, if the transcription for an image is already available in the original encoding rather than in Byblos

transliteration (as in, say, the case of a book for which both scanned page images and an HTML transcription are online), then a starting point for the transliterated transcription can be created automatically. In this case, the transcriber simply has to "correct" the automatically created transcriptions by inserting line breaks at appropriate locations in the text file.

4 Training







4.1 Pre-processing

Once the training images have been gathered, they are pre-processed (automatically) to remove skew. We then run the de-skewed images through our HMM-based line finding program that locates each line of text on the image. Since the line finding procedure may make errors, especially on noisy data, copies of the images with the line locations found can be viewed manually, and the line finding (only on training images and not the images held out for testing) corrected if necessary to match the transcriptions. Alternatively, if the line locations in an image have been marked separately, or if the image has already been broken into lines, that information can be used by the training system directly instead of performing line finding. With each line of the image found and de-skewed, a feature extraction program computes a feature vector sequence corresponding to that line [1, 2]. Given each feature vector and its matching transcription, model training can begin.

4.2 Training Algorithm

Our OCR system models each character with a multi-state, left-to-right HMM; the model for a word is the concatenation of the models for the characters in the word. Each state has an associated output probability distribution over the features. Each output probability distribution is modeled as a weighted sum of Gaussians, or what is called a *Gaussian mixture*. A Gaussian mixture is completely parameterized by the means and variances of the component Gaussians, along with the weight of each Gaussian in the mixture. The number of states and the allowable transitions are system

Table 1: Different representations for some sample Pashto glyphs.

| glyph |  |  |  |  |  |  |
|--|---|---|---|--|---|---|
| transliteration | h0 | h1 | h2 | h3 | n0 | N0 |
| character | HEH | HEH | HEH | HEH | NOON | NOON WITH RING |
| Glyph shape | Isolated | Initial | final | medial | isolated | isolated |
| code point | U+fee9 | U+feeb | U+feea | U+feec | U+fee5 | U+ea1a* |
| * Not an official Unicode code point; font's private-area mapping used | | | | | | |

parameters that can be set. For our experiments we have used 14-state, left-to-right HMMs with the topology shown in Figure 2.

Training – the process of estimating the parameters (transition probabilities and feature probability distributions) of each of the character HMMs – is performed using what has been known alternately as the Baum-Welch [3], forward-backward, or expectation-maximization (EM) algorithm [4,5], which iteratively aligns feature vectors with the character models to obtain maximum likelihood estimates of HMM parameters. The algorithm is guaranteed to converge to a local maximum of the likelihood function. The feature probability distributions in our system are characterized by the means, variances, and weights of the Gaussian mixtures.

The training process is performed as follows. We assume that, for each line of text, we are given the corresponding ground truth, which is simply the sequence of characters on that line. Note that no information is assumed about the location of each character on the line; that is, no pre-segmentation is necessary. The training algorithm automatically and iteratively aligns the sequence of feature vectors along the line of text with the sequence of character models. This is why this method does not care whether the characters are connected or not and why it can handle languages with connected scripts as well as other scripts where the characters are not connected. It is also why touching characters due to fax or other degradations do not require any special handling with this method.

The language model, which provides the probability of any character sequence, is also estimated from the same corpus. Note that the text corpus here does not require the presence of corresponding images; only the sequence of characters in the text is needed. Because it is much less effort to gather a body of text than to transcribe images, it is feasible to gather a much larger set of data for training language models than for training character HMMs. The more data gathered, the better the estimate of each character's probability that can be obtained for the language of interest.

4.3 Sub-characters

For ideographic languages, such as Chinese, we have innovated the use of sub-characters to speed up the recognition process. As the name indicates, sub-characters represent portions of “whole” characters. Each character in an ideographic script can be modeled as a sequence of sub-characters, just as each word in an

alphabetic script can be modeled as a sequence of characters. However, while the character sequence that makes up a word is known *a priori*, the sub-character sequence that makes up an ideograph must be discovered. The BBN Bybloz OCR system accomplishes this discovery automatically, through an identification process described in [6]. In the case of Chinese, we have been able to use 1000 sub-characters to spell approximately 4000 whole characters without any significant loss in accuracy but at a significant gain in terms of recognition speed.

Apart from introducing an extra intermediate step for automatic identification of sub-characters, the rest of the training procedure stays the same. As well, the language model is trained on whole character sequences and not sub-character sequences.

4.4 Importance of Data Variety

In order for the resulting models to be sufficiently robust, a suitable variety and amount of data must be used. If a useful model for a glyph is to be trained, then the glyph must appear multiple times in the training data. As a general rule the more the training data the lower the recognition error rate. At the same time, it is important that the data contain a broad variety of fonts, style, quality, and data sources (faxes, photocopies, typewritten, etc.) to ensure the best performance on a new test set which might be from any one of those sources. It is important to note that variety in terms of graphical forms (fonts, serif) is as important as variety in terms of the source of data (fax, photocopies), as illustrated by the following experimental results.

Table 2 illustrates the importance of including data of different types for training recognition models by using the example of fax and clean (non-fax) versions of the same set of text images. First, we created two sets of training data. One set consists of “clean” text images from the University of Washington corpus. The second set of images was created by printing out the images in the clean training set, faxing them from one fax machine to another, and scanning the received fax documents into black-and-white computer bitmaps. We then ran three different training experiments: one using just the clean set, one experiment using the fax data set, and a third one containing both clean and fax data. Shown in Table 2 are the character error rates under the various conditions of training and test. As can be seen from the numbers in Table 2, the best overall performance is obtained from the model trained on a mix of fax and clean data.

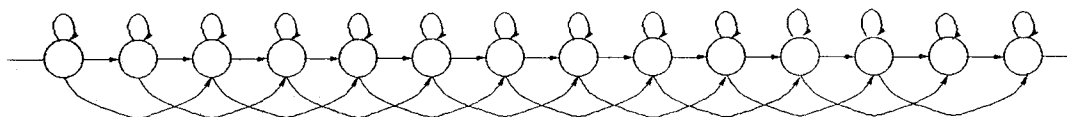


Figure 2: Topology

Table 2: English OCR performance - Clean vs. Noisy, fax data.

| Training | CER (Clean Test) | CER (Fax Test) |
|-------------|------------------|----------------|
| Clean Only | 0.6% | 5.3% |
| Fax Only | 1.0% | 2.2% |
| Clean + Fax | 0.6% | 2.7% |

Also important in order to model variability is the distribution across text style of the training data. The English OCR results in Table 3 illustrate the importance of ensuring appropriate representation for different styles/fonts in the training data. When our OCR system is trained on 600,000 characters of plain (non-italicized text in multiple fonts) and 6,000 characters of italicized text, the error rate on italicized test data is very high, about 25%. When an italics-only model is trained on 50,000 characters of italicized data, the error rate on italicized test data is 3.53%. A similar experiment on plain-text data shows that the error rate on plain text is 2.12% – a result that indicates that italicized data is inherently harder to recognize than plain text. Finally, the results in the last row of Table 3 clearly demonstrate that the best performance across styles is obtained by choosing a training data set that contains equal amounts of plain and italicized text. Other experiments have shown a similar sensitivity to font. Since for many languages only a few font families are in common use, it is often possible to include at least some data from each of the font families.

5 Optimizing System Performance

The first step in performing recognition is to run the pre-processing and feature extraction processes. After pre-processing a line of text and performing feature extraction, the recognition process consists in a search for the sequence of glyph models that has the highest probability of having generated the observed sequence of feature vectors, given the trained glyph models, a possible word lexicon, and a statistical language model of the possible character or word sequences. The recognition search is a two-pass (a forward pass and a backward pass) beam search for the most likely sequence of characters [8, 9]. The width of the search beam is a system parameter that can be set. Typically,

lowering the beam width increases the speed but degrades the accuracy of recognition. The forward pass is an approximate but efficient procedure for generating a small list of character sequences that are possible candidates for being the most likely sequence. The backward pass is a more detailed search for the most likely character sequence within this small list.

The width of the recognition beam is only one of the parameters that affect the performance of the recognition system. In order to obtain the best possible recognition performance using a given set of models, it is important to find the right set of weights that optimizes system performance. Using Powell's algorithm for finding local minima, the output from the recognition pass is used to search through the space described by a set of weights controlling the recognition search and find the "optimal" operating point that minimizes the error rate.

It is recommended that the optimization be rerun for each new type of data in order to obtain the best possible performance. The tuning procedure can be easily executed by transcribing a very small amount of the test data from a new source. Using the new weights obtained this way will boost performance during recognition. Optimizing the weights for the recognition search is the simplest form of system tuning and often yields significant improvements in performance. But if the test data is significantly different from the data used during training then more sophisticated approaches such as adaptation or even retraining should be considered. The use of adaptation for OCR is described in reference [7].

Table 4 illustrates the dramatic nature of the improvements that can be achieved with even limited amounts of adaptation or training data. The experimental results shown in Table 4 are on Pashto data. Because the work of adding support for Pashto recognition to our OCR system is very recent, our current models are trained on synthetic (computer generated) training data. Using these synthetic-data models for recognition results in a high 23.0% character error rate on scanned images of textbook pages. Unsupervised adaptation [7], which does not require any manual annotation or intervention, lowers the error rate to 20.5%. But retraining the models even with just one page of the textbook data added to the training data

Table 3: Character error rates for English OCR

| Number of Characters in Training Set | | Character Error Rate (CER) | |
|--------------------------------------|---------|--------------------------------------|---------|
| Plain | Italics | Plain (with a smattering of italics) | Italics |
| 600,000 | 6,000 | 1.19% | ~25% |
| | 50,000 | | 3.53% |
| 50,000 | | 2.12% | |
| 50,000 | 50,000 | 2.22% | 3.96% |

set cuts the error rate by half, down to 11.5%!

Table 4: Effect of in-domain training data on character error rates.

| Recognition models | Test Set | CER (%) |
|-------------------------------------|----------|---------|
| Synthetic | Textbook | 23.0 |
| Synthetic + unsupervised adaptation | Textbook | 20.5 |
| Synthetic + 1 page textbook data | Textbook | 11.5 |

6 Experimental Results

The BBN Byblos OCR system provides a truly language-independent recognition methodology. Using the procedure described in this paper, we have trained and tested our OCR system on several languages such as English, Arabic, Chinese, and Pashto.

For our English work, we used data from the University of Washington corpus, achieving character error rate (CER) of 0.6% on clean data and 2.7% on faxed data. In the case of our Arabic experiments, we used data from the DARPA Arabic corpus, with a CER of 2.89%. Due to the lack of a suitable public corpus for Chinese, we used data that was collected in-house. Our Chinese corpus consists of a large number of computer generated text images in a variety of fonts as well as a smaller set of "real world" documents consisting of about 16,000 characters of text images scanned from newspapers and magazines. Using a model trained on a mix of the synthetic and real data, a CER of 1.2% was obtained on a test set consisting of 8,000 characters of "real world" data.

Recently, we added support for the Pashto language that is used in parts of Afghanistan and Pakistan. The script for Pashto is very similar to Arabic but includes additional characters distinguished in some cases only by a small loop or tail from their Arabic equivalent, and is customarily printed in a different style of font. Initial results on the Pashto system show an error rate of 2.2% on synthetic data and about 3.5% on scans of textbook pages.

7 Conclusion

In this paper, we detailed our approach to porting the HMM-based Byblos OCR system to new languages or scripts and for improving performance on degraded data such as faxed documents. We demonstrated that, with the availability of appropriate training data, where ground truth is aligned only at the line level, the OCR system can be trained to recognize the new language with high accuracy. The system has been shown to

perform well for English, Arabic, Chinese, and Pashto, and for faxed documents.

References

- [1] P. Natarajan, Z. Lu, I. Bazzi, R. Schwartz, and J. Makhoul, Multilingual Machine Printed OCR, *International Journal of Pattern Recognition and Artificial Intelligence*, **15:1** (2001) 43-63.
- [2] I. Bazzi, R. Schwartz, and J. Makhoul, An Omnifont Open-vocabulary OCR System for English and Arabic, *IEEE Trans. Pattern Analysis and Machine Intelligence*, **21: 6** (1999) 495-504.
- [3] L.E.Baum, An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes, *Inequalities*, **3** (1972) 1-8.
- [4] A.P. Dempster, N.M. Laird, and D.B. Rubin, Maximum-likelihood from incomplete data via the EM algorithm, *J. Royal Statist. Soc. Ser. B (methodological)*, **39** (1977) 1-38.
- [5] R.A. Redner and H.F. Walker, Mixture densities, maximum likelihood and the EM algorithm, *SIAM Review*, **26** (1984) 195-239.
- [6] Z. Lu, R. Schwartz, P. Natarajan, I. Bazzi, and J. Makhoul, Advances in the BBN BYBLOS OCR System, in *Proc. of Intl. Conf. Doc. Analysis and Recognition*, Bangalore, India, 1999, 337-340.
- [7] P. Natarajan, I. Bazzi, Z. Lu, J. Makhoul, and R. Schwartz, Robust OCR of Degraded Documents, in *Proc. of Intl. Conf. Doc. Analysis and Recognition*, Bangalore, India, 1999, 357-361.
- [8] S. Austin, R. Schwartz, and P. Placeway, The Forward-Backward Search Algorithm, *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Toronto, Canada, 1991, Vol. V, 697-700.
- [9] R. Schwartz, L. Nguyen, and J. Makhoul, Multiple-Pass Search Strategies, in *Automatic Speech and Speaker Recognition: Advanced Topics*, C-H. Lee, F.K. Soong and K.K. Paliwal, eds. (Kluwer Academic Publishers, 1996) 429-456.

Segmenting and Tagging Structured Content

H. Ma, B. Karagol-Ayan, D. Doermann
 Institute for Advanced Computer Studies (UMIACS)
 University of Maryland, College Park, MD 20742

ABSTRACT

Bilingual dictionaries hold great potential as a source of lexical resources for training automated systems for optical character recognition, machine translation, and cross-language information retrieval. More importantly, they represent a class of document that have a great deal of structure, and this structure is not fundamentally spatial. Structure is provided by the authors (or publishers) who use of different fonts, font style, spacing and special symbols to implicitly convey information on the structure of the content.

Our system is divided into three phases – Dictionary Segmentation, Entry Tagging and Generation. In segmentation, pages are divided into logical entries based on structural features learned from selected examples. The extracted entries are associated with functional labels and passed to a tagging module that associates linguistic labels with each word or phrase in the entry. The output of the system is a structure that represents the entries of the dictionary.

In this document, we discuss the fundamental image processing approach that lets us extract this structure. Details of how we perform content tagging is mentioned briefly and references to related publications are provided.

1 Introduction

1.1 The Problem

In recent years, the demand for tools capable searching and retrieving written and spoken sources of multilingual information has increased tremendously. In this project, we are focusing on resource acquisition from bilingual dictionaries where the ultimate goal is the rapid training and realization of CLIR systems for low-density languages. Given a bilingual dictionary, with one of the two languages, English, a scanner, an optical character recognition (OCR) system that is representative of the character set, and an operator familiar with the language in question, we hope to be able to train a retrieval system for the new language in as little as 48 hours. To do this, we need to provide an automated, but user guided approach, to parameterize and learn the physical structure of the document page, and the semantic of the dictionary entries.

A key observation is that this type of processing is applicable to many other document analysis tasks. There are often times where system are designed to handle very general documents, but where imposing structural constraints via simple example can significantly improve results.

1.2 Background

Dictionaries come from a class of documents that are designed for easy search [DOE 98b]. The structure is typically regular and repeating, and “keys” are distinguished as access points for each entry, definition or translation. Figure 1 shows some

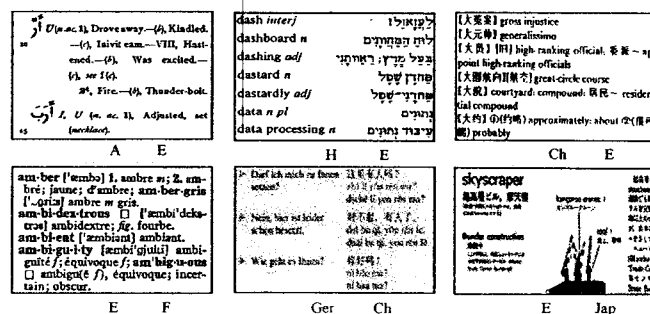


Figure 1: Examples of Bilingual Dictionaries.

common formats of bilingual dictionaries. The format varies from simple term and phrase translation pairs to full descriptions that contain parts of speech, forms and examples of usage. We need to capture the salient structure of these entries and label each element appropriately. Because of the regular structure, we should be able to provide a relatively small number of training samples for each dictionary, and have the system learn the features necessary for correct segmentation and labeling.

1.3 Approach

A prototypical architecture for a system to generate cross language term lists is shown in Figure 2. The system is broken up into three main components: Dictionary Parsing, Element Tagging and Lexicon generation. The general philosophy we take with all of the components is to provide a trainable system that can learn from a limited number of examples. An operator will identify and tag several dictionary entries, and it is the systems responsibility to learn the parameters necessary to extract and tag the remainder of the dictionary. Because of the repetitive nature of these documents, a bootstrapping approach where the system gets some feedback from an operator is appropriate. Although many challenge exist in producing a general document image analysis system to perform segmentation, our goal is to provide the tools to tune our algorithms to each dictionary, independent of the operator. Because of the nature of most dictionaries, it is essential to have an operator who can at least understand the both languages in the bilingual dictionary. We provide an intuitive interface to assist in processing the dictionary and compilation of results.

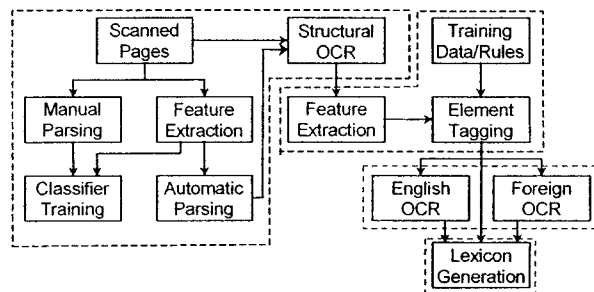


Figure 2: System Architecture.

2 Segmentation Overview

The first step in analyzing the dictionary involves physically segmenting the dictionary into individual entries, extracting functional characteristics of various elements of the entry and labeling the type extracted zones.

Entry segmentation addresses the general problem of identifying repeating structures by learning the physical and semantic features that characterize them. Unlike traditional page segmentation problems where zones are characterized by spatial proximity, we often find that publishers of documents with multiple entries (dictionaries, phone books, and other lists) use different text (bold, italics, font size etc) and layout (indentation, bullets, etc) features to indicate a new entry. Although such characteristics vary for different documents they are often consistent within a single document, and hence the task suggests learning techniques.

The second task involves extracting functional properties of words or groups of words that publishers use to make semantic labels implicit. For example, changes in font-face or font-style are often used to represent the linguistic meaning of various words such as pronunciations, parts of speech or examples of usage. At a high level, working with non-Latin documents requires us to be able to identify scripts before applying an appropriate OCR algorithm. These types of features prove to be extremely useful for both parsing and tagging.

The final task involves labeling the type for each entry extracted. An entry may extend across columns or zones while other entries may need to be ignored because they are not of interest for logical labeling (for example, a page number, header or footer). The type is passed along to the tagging module.

2.1 Related Work

Work relevant to entry segmentation is typically found in the logical layout analysis literature ([LEB 01] gives an overview of traditional methods). The process of document layout structure analysis is often divided into two tasks: physical segmentation and logical labeling. Physical segmentation divides a page into zones with specific physical characteristics. Logical analysis labels each extracted zone with a specific functional or logical label. The structural complexity of different documents makes it difficult to design a generic document analysis tool that can be applied to all documents. Furthermore, since logical analysis is often based on the physical segmentation result, the performance of the physical segmentation module is crucial for understanding the document image and dominates performance.

Liang et al. [LIA 01] presented a probability-based text-line identification and segmentation approach. Their approach consists of two phases: an offline statistical training and online text-line segmentation. In the online text-line segmentation phase, an iterative, relaxation-like method was applied to find an optimal partition of source entities by improving a conditional probability. Kopec et al. [KOP 94] applied a stochastic approach to build Markov source models for text-line segmentation under the assumption that a symbol template is given and the zone (or text columns) had been previously extracted. Under the assumption that the physical layout structures of document images can be modeled by a stochastic regular grammar, Mao et al. [MAO 01] used a generative stochastic document model to model a Chinese-English dictionary page, and a weighted finite state automaton that model the projection profile at each level of document physical layout tree is used to segment the dictionary page on all levels. Lee et al. [LEE 01] proposed a parameter-free method to segment document images with various font sizes, text-line spacing and document layout structures. There are also some rule-based segmentation methods which perform the segmentation based on rules that are either manually set up by user [LEB 01] or learned automatically by training [MAL 01][KOP 94].

In work on script identification, Hochberg et al. [HOC 97] described a technique for identifying 13 scripts including highly connected ones. In their algorithm, a scale-normalized cluster template is created for each script based on the frequent characters or word shapes of this script, then scripts are classified by comparing a subset of the document's textual symbol with these templates. In [SPI 97][SIB 94], Spitz et al. initially divided scripts into Asian (Chinese, Japanese and Korean) and Roman based on the observation that upward concavities are distributed evenly along the vertical axis of Asian characters, but tend to appear at certain locations in Roman characters. Further distinctions among Asian scripts are made on the basis of character density (Figure 3). In their work, Waked et al. [WAK 98] used the different features of horizontal projection of text line to classify scripts into three categories including Arabic, Roman and Ideographic scripts.

For font recognition, Manna et al. [MAN 99] used the tangent distance as a classification function in a nearest neighbor approach. In order to improve the performance, a TD-Neuron discriminant model is employed to discriminate between two similar classes. To classify italic text, Kavallieratou et al. [KAV 01] presented a slant removal algorithm based on the vertical projection profile of word images and the Wigner-Ville distribution and Vinciarelli et al. [VIN 00] presented an algorithm that detects the slant based on the number of vertical strokes. Zramdini et. al. presented an optical font recognition approach using typographical features,

| |
|---|
| <p>ad·vice [əd'vaɪs] <i>n.</i> ① 忠告, 劝告; 建议; 指教. ② (医生等的) 诊察, 意见. ③ [常 <i>pl.</i>] (政治, 外交上的) 报导, 报告. ④ [商] 通知. <i>an ~ note</i> 通知单. <i>a remittance ~</i> 汇款通知. <i>~s from foreign countries</i> 来自国外的报导. <i>a written ~</i> 劝告书. <i>act on ~</i> 依劝. <i>ask ~ of</i> 向...征求意见, 请教, 领教. <i>by [on] sb.'s ~</i> 依某人劝告. <i>follow sb.'s ~</i> 接受某人意见. <i>give [tender] ~</i> 劝告, 忠告. <i>take ~</i> 征求意见, 请教, 领教 (<i>take medical ~</i> 请医生诊视). <i>take sb.'s ~</i> = <i>follow sb.'s ~</i>. ad·vis·a·bil·i·ty [əd,vaɪzə'bɪlɪti] <i>n.</i> 可劝告; 适当, 得当.</p> |
|---|

Figure 3: English-Chinese Dictionary.

and the recognition is based on a multivariate Bayesian classifier and operate on a given set of known fonts. In recent years, texture analysis techniques have been introduced to classify different font-styles and font-faces. Zhu et al. presented a font recognition algorithm based on global texture analysis, where Gabor filters were used to extract the global texture features.

3 Entry Segmentation

3.1 Overview

The goal of entry segmentation is to segment each zone into multiple entries or organize multiple text-lines into one entry. Since the extraction of text lines in dictionaries is relatively straight forward, the problem of entry segmentation can be posed as the problem of finding the first or last lines of an entry. We use an iterative procedure that maximizes the feature voting score of an entry in the feature space. This search operation is a threshold-based and relaxation-like procedure, and the threshold is estimated from a small training set (as few as 2-3 entries). Considering the fact that there are a relatively small number of text lines in one page, this search can be done by brute force. The segmentation procedure is described as:

- (1) Search candidates for the first text line in one zone by feature matching. This operation is equivalent to determining if the first line in one zone is the beginning of a new entry or a continuation of an entry in the previous zone or previous page.
- (2) Search for the end of an entry. This operation is replaced by searching the beginning of next entry because the beginning of a new entry is the ending of the previous entry.
- (3) Remove the extracted entries and iterate until all new entries are identified.

A significant contribution to the entry segmentation is the application of a bootstrap technique for the generation of new training samples. Bootstrapping helps to make the segmentation adaptive and progressively improve the segmentation performance. Figure 4 illustrates the approach with each iteration consisting of the following three steps:

Feature Extraction: The segmentation system is automatically trained using a small set of labeled samples and features of entries are extracted.

Segmentation: Pages are segmented based on the extracted features.

Correction and Bootstrapping: The segmented results are fed back to an operator, who makes corrections to a small subset of the results with errors. Using the corrected segmentation results, bootstrapping samples are generated and used to retrain the system.

3.2 Feature Extraction

Based on a study of different types of structured documents, the following features have been shown to be useful for both segmentation and ultimately for tagging. Examples are shown in Figure 5.

- **Special symbols:** Punctuation, numbers, and other non-alphabet symbols are often used to start a new entry, to end an entry, or to mark the continuation of a text line.

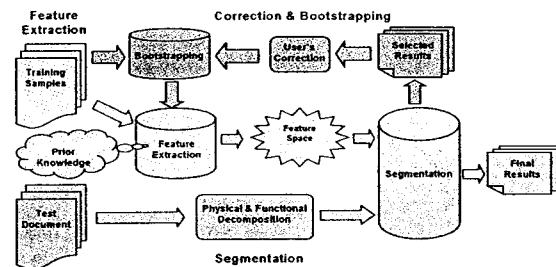


Figure 4: Entry Segmentation System Architecture.

- Word font, face and size: Word font, face and size (especially the features of the first word in each entry) are often important entry features. On a dictionary page, for example, the first word of each entry (typically the headword) can be bold, all capital, a different font, or larger than the rest of the entry.
- Word patterns: Words often form distinguishable patterns which can be used to describe the entry structure consistency.
- Symbol patterns: Combined with other symbols or regular characters, special symbols can form some consistent patterns to represent the beginning or ending of entry.
- Line structures: The indent, spacing, length, height of text lines in an entry all can be contained in the line structures to represent the features of an entry.
- Other features: Other features can also be used to segment the entries such as spacing between adjacent entries, the position of text, script type, word spacing, character case and so on.

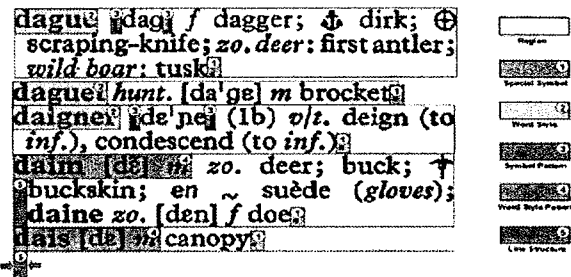


Figure 5: Example of Features.

It should be noted that some of the features, such as special symbols, symbol patterns and character case, cannot be obtained without first performing OCR. Some of the features (word font, font-face, word pattern, script type etc.) are available only after further classification. The identification of scripts, font and font-faces can significantly improve the performance of the OCR, and the performance of OCR will affect the performance of segmentation. The OCR therefore could be performed either directly on the scanned images (for example if the document contains only one script), or on the scanned image after script, font-style and font-face classification (Figure 2). In this section, we first assume the OCR results, including font and font-face identification, are available. The identification of scripts, font-styles and font-faces will be described in Section 4.

During the training phase, a Bayesian framework is used to assign and update the probabilities of extracted features. Based on estimated probabilities, each extracted feature will be assigned a weight that can be used to compute the entry score from all extracted features. The detailed procedure is as follows:

- 1) Construct a histogram of feature occurrences in the training samples;
- 2) Compute feature occurrence rate as the feature probability. Suppose there are N training entries, and there are K extracted features. Then for feature i ($1 \leq i \leq K$), the probability can be computed as: $p_i = K_i/N$, where K_i is the number of occurrence of feature i .
- 3) Assign feature weights based on the computed probability as follows:

$$w_i = \frac{P_i}{A} \times 100 \quad \text{where } A = \sum_{i=1}^K p_i \quad \text{and } i \leq K$$

Consider the extracted features as a formed feature space, each entry is projected to this space and a voting score is computed as follows:

$$FV = \sum_{i=1}^K w_i S_i, \quad \text{where } S_i = 1 \text{ if the feature } i \text{ occurs, otherwise } S_i = 0.$$

- 4) Obtain the minimum, maximum, average voting scores of entries, these values will be used as thresholds in the segmentation stage.

Different types of entries may have different feature occurrences, so this procedure is run for each type of entry and we scale the values to facilitate computation. Figure 6 shows an example of

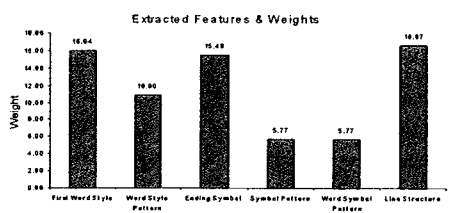


Figure 6: Extracted Features & Weights Corresponding to Dictionary Shown in Figure 5.

extracted features with assigned weights. We can see that for this example the line structure (negative indent of first text-line) has the heaviest weight, which means it is the most important feature in this document.

3.3 Training and Segmentation

Because of the complexity of many structured documents, it is difficult to determine the optimal value of some parameters. So given a small training set, we attempt to learn them. One way to do this is to extract all possible features from the training set, segment some pages based on the learned features, and generate a new training set from the original set and selected initial segmentation results. This technique used to generate new “bootstrap” samples is the so-called bootstrap technique that was first proposed by Efron [EFR 79].

Before combining the segmentation results with original training samples to generate bootstrap samples, the operator makes corrections to the original segmentation results by performing one or more of the following operations:

- Split: split one segmented entry into two or more individual entries
- Merge: merge two or more adjacent entries into one single entry
- Resize: change the size of a segmented entry
- Move: change the bounding box position of a segmented entry
- Remove: remove a segmented entry
- Relabel: change the type label of an entry

We applied a procedure to generate bootstrap samples, and the sample entries are chosen such that no entry is selected more than once. Generated bootstrap samples are the linear combination of the training samples in source based on random weights. Additional details of iterative process can be found in [MA 03a]

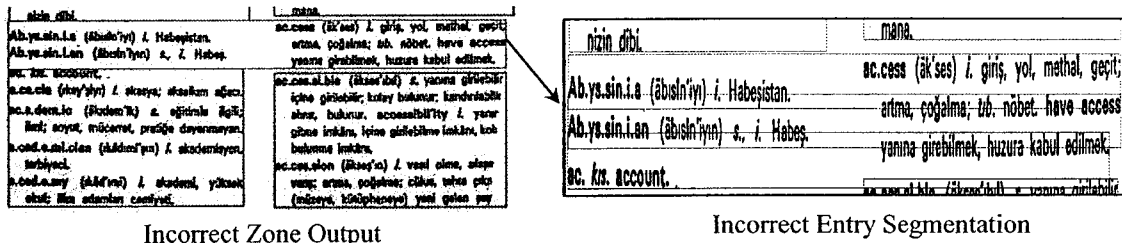


Figure 7: Segmentation Errors Caused by Zoning Error.

3.4 Post-Processing

Entry segmentation result is heavily dependent on the zone segmentation results. In other words, if the zone segmentation result is incorrect, it is impossible to obtain correct entry segmentation results from the zone segmentation without adjustment (Figure 7). So the task of the post-processing is to automatically correct the incorrect entry segmentation results caused by incorrect zone segmentation and make the segmentation approach more adaptive. The post-processing procedure illustrated in Figure 8. Statistical information extracted includes: *average width of entry*, *average text-line height of entry*, *average regular text-line width*, *word spacing within entry*, *relative position of interested entry* and so on. We browse the words in each of entries with significant difference from the statistical information obtained from training samples and reorganize words into text-lines and entries as appropriate. Figure 9 shows the segmentation result after post-processing.

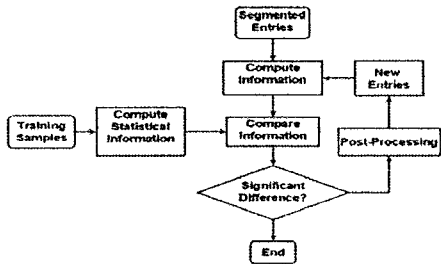


Figure 8: Post-processing Flowchart.

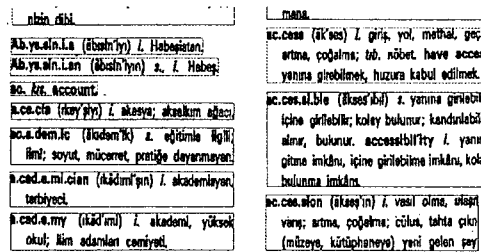


Figure 9: Correct Result.

4 Functional Labeling

It should be noted that all the script identification approaches mentioned in the related work are at the block or page level, and the font recognition method in [ZHU 01] requires attempt to identify the “primary” font on the page. Obviously, this is not the case for bilingual dictionaries where words in different languages can be interspersed and so script identification must be done at the word level. In our work, we perform classification based on the Gabor filter analysis of textures of different classes.

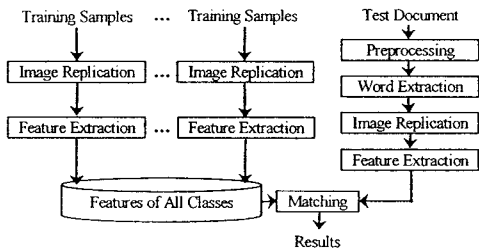


Figure 10: Classification System.

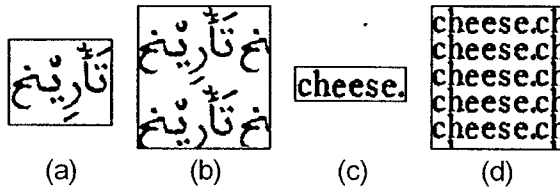


Figure 11: Examples of Image Replication. (a)(c): Before; (b)(d): After.

The use of Gabor filters in extracting texture features of images is motivated by factors that: (1) Gabor representation was shown to be optimal in the sense of minimizing the joint two-dimensional uncertainty in space and frequency [DAU 88]; and (2) Gabor filters can be considered as orientation and scale tunable edge and line detectors, and the statistics of these micro-features in a given region are often used to characterize the underlying texture information. The system to classify scripts, font-styles and font-faces is shown in Figure 10.

We create a set of training samples for each class by randomly picking words from a large set of words that belong to that class. Different words in different scripts, different font-styles and different font-faces in the same script have different dimensions (width & height), so we use word image replication to generate a new image with predefined size (64x64 pixels in our case), all features are extracted on the same image size. For computation, the replication is to a power of two such that FFT can be applied in computation of Gabor filter features. Figure 11 shows word replication examples of two different scripts (Arabic, English). After generating a word images, multi-channel Gabor filter technique was applied to extract the texture features of each class. We use the same pairs of isotropic Gabor filters as in to extract the texture information.

Based on the image size (64x64), four values of spatial frequency are selected: 0.04, 0.08, 0.16 and 0.32. These four frequencies combine with four angles θ ($0^\circ, 45^\circ, 90^\circ, 135^\circ$) give a total of 16 Gabor channels.

After extracting channel features, the matching (classification) procedure based on feature vector is a typical pattern recognition problem, and distance between two image patterns is used in the classification. Details of the computational models used as well as feature representation can be found in [MA 03b].

5 Experiments

5.1 Segmentation Results

The segmentation approach was applied to four different dictionaries with different structure features: French-English dictionary (613 pages), English-French dictionary (657 pages), Turkish-English dictionary (909 pages), English-Turkish dictionary (1152 pages). Figure 12 shows the result of English-French dictionary.

| 632 | 632 | 632 |
|---|---|---|
| <p>an·cient ['eɪnfənt] 1. ancien(ne <i>f</i>); antique; 2. the <i>s</i> pl. les anciens <i>m/pl.</i> (grecs et romains); 'an·cient·ly anciennement; jadis.</p> <p>an·cil·lar·y [æ'n'siləri] <i>fig.</i> subordonné; ancillaire (<i>à, to</i>); accessoire (<i>à, to</i>).</p> | <p>an·cient ['eɪnfənt] 1. ancien(ne <i>f</i>); antique; 2. the <i>s</i> pl. les anciens <i>m/pl.</i> (grecs et romains); 'an·cient·ly anciennement; jadis.</p> <p>an·cil·lar·y [æ'n'siləri] <i>fig.</i> subordonné, ancillaire (<i>à, to</i>); accessoire (<i>à, to</i>).</p> | <p>an·cient ['eɪnfənt] 1. ancien(ne <i>f</i>); antique; 2. the <i>s</i> pl. les anciens <i>m/pl.</i> (grecs et romains); 'an·cient·ly anciennement; jadis.</p> <p>an·cil·lar·y [æ'n'siləri] <i>fig.</i> subordonné, ancillaire (<i>à, to</i>); accessoire (<i>à, to</i>).</p> |
| Word | Text-line | Entry (page no is noise) |

Figure 12: Entry Segmentation Result of English-French Dictionary.

Figure 13 shows the performance improvement based on bootstrapping. The results evaluation comes from the statistical information of 50 pages of each dictionary. The initial segmentation was based on only four training samples. Iterations following the initial segmentation are based on adding different numbers of training entries which are used to generate bootstrap samples. The chart in Figure 13 shows that the segmentation can be refined step by step by applying the bootstrap technique.

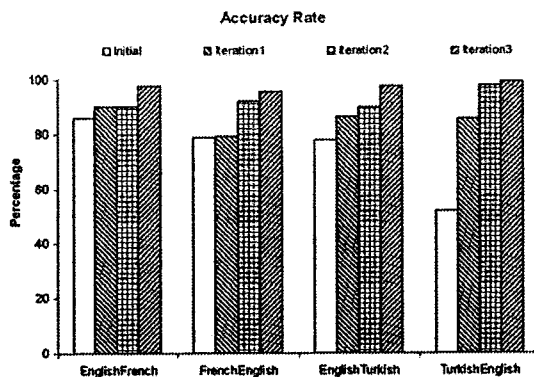


Figure 13: Progressive Performance Improvement.

Table 1: Segmentation Results of Four Dictionaries.

| Document | Page No | Total Entries | Correct Entries | Incorrect Entries | | | | False Alarm | Mislabelled Entries |
|----------------|---------|---------------|-----------------|-------------------|------------|--------|-------|-------------|---------------------|
| | | | | Missed | Overlapped | Merged | Split | | |
| EnglishFrench | 635 | 20174 | 96.11% | 0.005% | 1% | 2.62% | 0.26% | 0.21% | 0.8% |
| FrenchEnglish | 75 | 2423 | 97.9% | 0.04% | 1.61% | 0.25% | 0.21% | 0.25% | 0.49% |
| EnglishTurkish | 96 | 3517 | 99.26% | 0.0% | 0.17% | 0.11% | 0.45% | 0.23% | 0.31% |
| TurkishEnglish | 70 | 2654 | 98.98% | 0.04% | 0.26% | 0.0% | 0.72% | 0.08% | 0.38% |

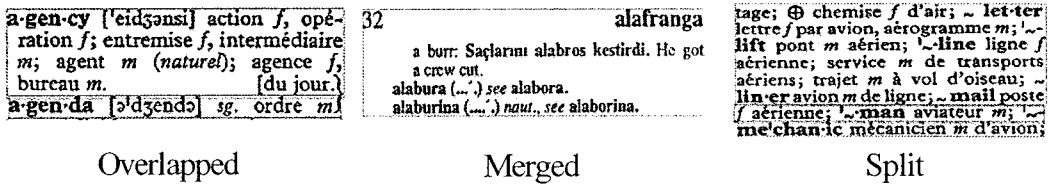


Figure 14: "Incorrect Entry" Errors.

The performance evaluation is shown in Table 1, which was based on the available ground truths of these dictionaries. In Table 1, the "Correct Entries" and "Incorrect Entries" are two complementary parts of the evaluation results, whose sum is 1. The four different error instances are shown in Figure 14. "False Alarm" error measures the impact of noise on the segmentation result, and "Mislabelled Entries" errors measure incorrect labeling of segmented entries. We can see the algorithm works well such that the lowest percentage of correct segmentation is higher than 96%, and the best result can even achieve higher than 99%

5.2 Script Identification and Font Recognition Results

We applied the presented script identification approach to the Arabic, Korean and Chinese-English bilingual dictionaries. Each script class's features were extracted from 10 randomly chosen samples. Figure 15 shows the identification results, and the script identification evaluation is shown in Table 2 (where the results are based on one page classification).

The font-face classification approach trained using 10 samples was applied to the identified English (Roman) document images, and one of the identification results of Korean-English dictionary is shown in Figure 16, the evaluation of this result is shown in Table 3.

From experimental results shown in Figure 15, 16 and Table 2, 3, show that the Gabor filter based multi-class classifier at word level works effectively, the average script identification accuracy is higher than 86%, and the highest rate is close to 98%.

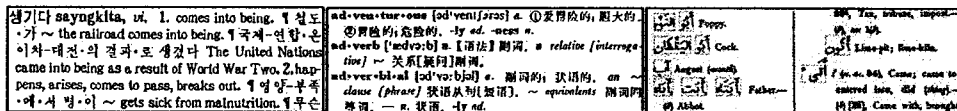


Figure 15: Identified Non-Roman Scripts.

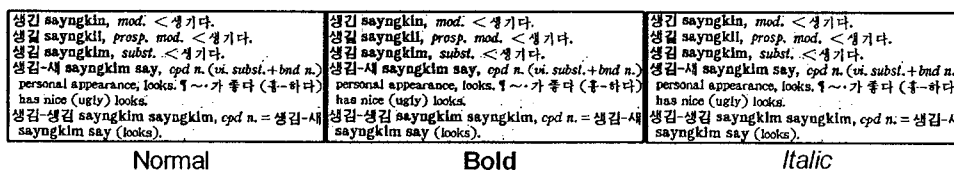


Figure 16: Classified Font-Faces of Roman Script.

Table 2: Script Identification Result.

| | Roman | | | Other Script | | |
|----|-------|-----|-------|--------------|-----|-------|
| | TN | ID | A (%) | TN | ID | A (%) |
| CE | 373 | 321 | 86.1 | 135 | 132 | 97.8 |
| KE | 334 | 292 | 87.4 | 202 | 183 | 90.6 |
| AE | 277 | 271 | 97.8 | 36 | 32 | 88.9 |

(CE: Chinese-English, KE: Korean-English, AE: Arabic-English, TN: Total Number, ID: Identified Number, A: Accuracy)

For the script classification, many of the errors in identification were caused by incorrect word segmentation. Sometimes two different word could be merged into one single word, so we strongly believe that the identification performance could be improved by improving the word segmentation performance.

The font-face classification is worst for “Normal” (73.6%) but much better for “Bold” and “Italic” (both are higher than 85%). By browsing the classification results, we found most of the incorrect classifications of “Normal” font-face were caused by low quality images, and some of them were also caused by the incorrect word segmentation.

The classifier was applied to an artificial test document image to show the effectiveness to classify different closely related font-faces. The test image was created to contain two frequently used Roman script fonts “Arial” and “Times New Roman”. In the image, words of these two fonts appear alternatively. The font-style classifier was trained by words taken from a different document (each with 10 training words). The classification results and evaluation are shown in Figure 17, and the evaluation is shown in Table 4. Word size plays a significant role so we need to consider the effect of the number of characters in the future.

6 Discussion

The focus of this paper is on the initial components required to segment and functionally label bilingual dictionary entries because of the obvious implications to more general document analysis problems. The ability to bootstrap recognition to tailor systems to process long documents with consistent style is an essential capability in many environment. We have shown the ability to train segmentation and functional labeling algorithms to adapt to variation in both content and visual appearance.

For bilingual dictionaries, the “entries” themselves are just as structurally rich. Furthermore, publishers typically use a combination of methods to indicate the linguistic meaning. Functional properties (changes in font, font style, font-size, etc) are used to make linguistic meaning implicit, *keywords* to make meaning explicit, and various *separators* are used to organize them. For instance, headwords may be in bold, examples of usage in italics; the keywords “*adj*”, “*n*” or “*v*” may be used to explicitly represent the part-of-speech, or we may find the pronunciation offset with brackets, commas used to separate different examples of usage, or a numbering system used to identify different tenses of the word. Details of the approach to tagging and generation can be found in [MA03c]

Table 3: Font-Face Classification Result of Roman Script.

| | Normal | Bold | Italic |
|----------------------|--------|------|--------|
| Total No | 250 | 40 | 44 |
| Identified No | 184 | 35 | 39 |
| Accuracy (%) | 73.6 | 87.5 | 88.6 |

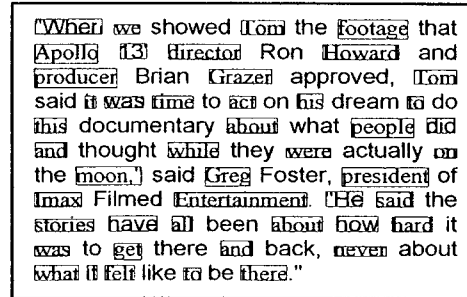


Figure 17: Identified Times Font of Roman Script.

Table 4: Result of Font Classification of Roman Script

| | TN | Times | Arial | Detection |
|--------------|----|-------|-------|-----------|
| Times | 39 | 39 | 7 | 100% |
| Arial | 39 | 7 | 32 | 82.1% |

7 Acknowledgements

The support of this project by DARPA under cooperative agreement N66001002891002, the National Science Foundation grant EIA0130422 and the Department of Defense under contract MDA90402C0406 is gratefully acknowledged.

8 Bibliography

- [DAU 88] J.G. Daugman, *Complete Discrete 2D Gabor Transforms by Neural Networks for Image Analysis and Compression*, *IEEE Trans. Acoustics, Speech and Signal Processing*, VOL. 36, July 1988, pp.1169-1179.
- [DOE 98a] D. Doermann, *The indexing and retrieval of document images: a survey*, *Computer Vision and Image Understanding*, Vol 70:3, 287-298, 1998.
- [DOE 98b] D. Doermann, E. Rivlin, and A. Rosenfeld, *The Function of Documents*, *IJCV*, 16, pages 799-814, 1998.
- [DOE 02] D. Doermann, H. Ma, B Karagol-Ayan, D. W. Oard, *Translation Lexicon Acquisition from Bilingual Dictionaries*, *Proc. SPIE Conf. Document Recognition and Retrieval*, 37-48, San Jose, CA, January, 2002.
- [EFR 79] B. Efron, *Bootstrap Methods: Another Look at the Jackknife*, *Annual Statistics* vol 7, 1-26, 1979.
- [HOC 97] J. Hochberg, P. Kelly, T.Thomas, and L. Kerns, *Automatic Script Identification From Document Images Using Cluster-Based Templates*, *IEEE Trans. Pattern Analysis and Machine Intelligence*, VOL. 19, NO. 2, February 1997, pp. 176-181.
- [KAV 01] E. Kavallieratou, N. Fakotakis, and G. Kokkinakis, *Slant Estimation Algorithm for OCR System*, *Pattern Recognition*, VOL. 34, No. 12, December 2001, pp. 2515-2522.
- [KOP 94] G. E. Kopec, P. A. Chou, *Document Image Decoding using Markov Source Models*, *IEEE Tran. Pattern Analysis & Machine Intelligence*, vol.16:6, 602-617, June, 1994.
- [LEB 01] F. LeBourgeois, S. Souafi-Bensafi, J. Duong, et. al. *Using statistical models in document images understanding*, *DLIA2001 Advance Program*, Seattle, WA, Sep. 2001.
- [LEE 01] S. Lee, D. Ryu, *Parameter-Free Geometric Document Layout Analysis*, *IEEE Tran. Pattern Analysis & Machine Intelligence*, vol 23:11, 1240-1256, November, 2001.
- [LIA 01] J. Liang, I.T. Phillips, R.M. Haralick, *An optimization Methodology for Document Structure Extraction on Latin Character Documents*, *IEEE Tran. Pattern Analysis & Machine Intelligence*, vol. 23:7, 719-734, July 2001.
- [MA 03a] H. Ma, D. Doermann, *Bootstrapping Structured Page Segmentation*, *Proc. SPIE Conf. Document Recognition and Retrieval*, Santa Clara, CA, January, 2003.
- [MA 03b] H. Ma, D. Doermann, *Gabor Filter Based Multi-class Classifier for Scanned Document Images*, submitted to 7th International Conference on Document Analysis and Recognition, Edinburgh, Scotland, August 2003.
- [MA 03c] H. Ma, B. Karagol-Ayan and D. Doermann. *Translation Lexicon Acquisition and Application from Bilingual Dictionaries*. *TAL Traitement Automatique Des Langues*, 2003 . (To Appear).
- [MAL 01] D. Malerba, F. Esposito, *Learning Rules for Layout Analysis Correction*, *DLIA 2001 Advance Program*, Seattle, WA, Sep. 2001
- [MAN 99] S.L. Manna, A.M. Colla, and A. Sperduti, *Optical Font Recognition for Multi-Font OCR and Document Processing*, 10th International Workshop on Database and Expert Systems Applications, Florence, Italy, 1-3 September, 1999, pp. 549-553.
- [MAN 99] C.D. Manning, Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press. 1999.

- [MAO 01] S. Mao, T. Kanungo, *Stochastic Language Models for Automatic Acquisition of Lexicons from Printed Bilingual Dictionaries*. DLIA2001 Advance Program, Seattle, WA, Sep. 2001.
- [SIB 94] P. Sibun, and A.L. Spitz, *Language Determination: Natural Language Processing from Scanned Document Images*, Proc. 4th Conference on Applied Natural Language Processing, Stuttgart, October 1994, pp. 115-121.
- [SPI 97] A.L. Spitz, *Determination of the Script and Language Content of Document Images*, IEEE Trans. Pattern Analysis and Machine Intelligence, VOL. 19, NO. 3, March 1997, pp. 235-245.
- [VIN 00] A. Vinciarelli, and J. Luetin, *Off-line Cursive Script Recognition Based on Continuous Density HMM*, Proc. 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, 11-13 September, 2000.
- [VOO 98] E.M. Voorhees, *Variations in Relevance Judgments and the Measurement of Retrieval Effectiveness*, Proc. 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August, 1998, pp. 315-323.
- [WAK 98] B. Waked, S. Bergler, and C.Y. Suen, *Skew Detection, Page Segmentation, and Script Classification of Printed Document Images*, IEEE International Conference on Systems, Man, and Cybernetics (SMC'98), San Diego, California, USA, October 1998, pp. 4470-4475.
- [ZHU 01] Y. Zhu, T. Tan, and Y. Wang, *Font Recognition Based on Global Texture Analysis*, IEEE Trans. Pattern Analysis and Machine Intelligence, VOL. 23, NO. 10, October 2001, pp. 1192-1200.

Handwriting

A System for Handwriting Matching and Recognition⁺

Sargur N. Srihari, Bin Zhang, Catalin Tomai, Sangjik Lee, Zhixin Shi and Yong-Chul Shin
CEDAR (Center of Excellence for Document Analysis and Recognition)
University at Buffalo, State University of New York
Amherst, New York 14228
srihari@cedar.buffalo.edu

ABSTRACT

Architecture and a system for computer analysis of scanned handwritten documents are described. Functionalities of the system are: (i) handwriting matching to determine whether a questioned handwriting sample matches a known handwriting sample, (ii) handwriting recognition for free form handwritten documents and (iii) a merger of the two so that recognition results can be used in writer identification and identification results can be used in handwriting recognition.

Key Words: *Writer Identification, Handwriting Recognition*

1. INTRODUCTION

Handwriting is a natural means of human communication and a method for recording small amounts of information by humans. Thus the processing of handwriting by computer has numerous applications such as improved human-machine interfaces. This research explores the relationship between two topics in processing handwriting by computer: handwriting recognition and writer identification. Handwriting recognition is the task of recognizing the message conveyed by handwriting. It is performed by recognizing the visual characteristics of the writing. On the other hand, the task of identification is to determine the writer from the idiosyncrasies of the writing.

Recognition involves averaging out the differences in different writers so that specific idiosyncrasies of the writer are averaged out. Identification involves understanding those very idiosyncrasies with the message content being a secondary issue. The key question in writer identification is to identify the individual while considering the fact that the same individual can write differently in different instance, i.e., to capture the inherent variance in the measurements for the same individual while differentiating between the measurements for different individuals.

⁺ This work was supported in part by the U.S. Department of Justice, National Institute of Justice grant 2002-LT-BX-K007.

Both of these tasks are illustrated in Figure 1 in their simplest form, where handwriting consists of simple hand printed letters. The figure contains a matrix of handwritten letters corresponding to five writers (each corresponding to a row of writing) writing four different letters (each corresponding to a column I, N, 0, and 1). Given a new character image whose identity and writer are unknown, the recognition task is that of determining the column (which character is it?) while identification is the task of determining the row (which writer is it?). If one were to determine both the row and the column then we have identified both the writer and the message (letter) conveyed.

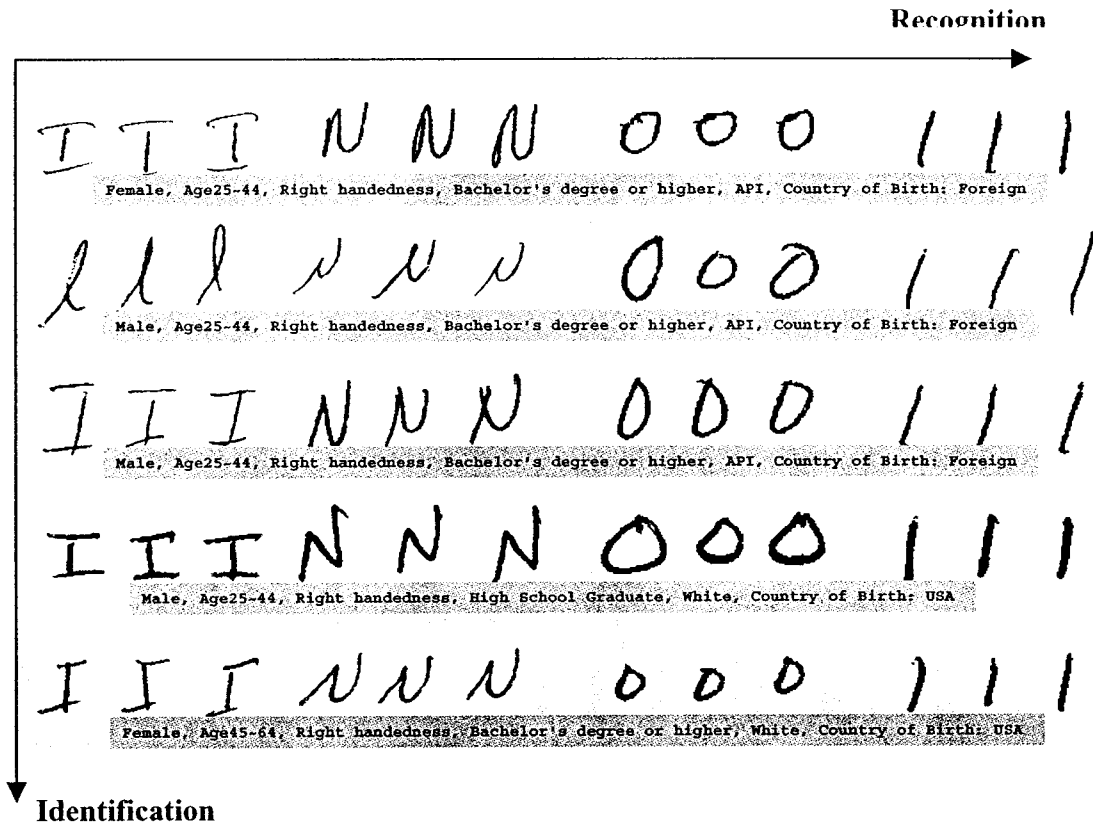


Fig. 1. A Recognition-Identification matrix. Each of five rows corresponds to a different writer. Columns correspond to each of them writing four characters, I, N, 0 and 1 three times each..

Since the same writer can write differently each time, there are component statistical distributions in feature space corresponding to each letter for each writer. The recognition task involves mixing the distributions along the columns of Figure 1 so that the individual characters can be identified. The identification task is to combine the letter-specific distributions along each row of Figure 1 to discriminate between individuals. The example of Figure 1 pertains only to hand printed characters while in

general the recognition and identification tasks extend over the full range of cursively written handwritten words and sentences.

There are several functionalities for a computer system for processing handwriting: (i) writer identification -- to retrieve the identity of a writer from a set of writers given a questioned document, (ii) handwriting matching -- determining whether two handwritten documents were written by the same individual, (iii) writer-independent handwriting recognition -- determining the message conveyed in handwriting using writer-independent character and word recognition methods, (iv) incorporating handwriting identification, or writer characteristics, in the task of recognition, and (v) incorporating recognition results in the task of identification.

2. PREVIOUS WORK

Our discussion of the relevant literature on automated methods is divided into two parts: handwriting recognition and handwriting matching/writer identification.

2.1 Handwriting recognition

Handwriting recognition has a significant-sized published literature, e.g., there have been eight International Workshops on Handwriting Recognition, which have proceedings. A comprehensive survey of both on-line and off-line handwriting recognition can be found in [Plamondon and Srihari, 2001].

There has been considerable success in constrained-domain handwriting recognition, e.g., in handwritten address interpretation [Srihari, et. al, 1989, Cohen, et. al, 1991, Srihari and Keubert, 1997, Mahadevan and Srihari, 1999]. More generally, a system for reading unconstrained handwritten pages known as PENMAN was developed by us [Srihari and Kim, 1997]. Other recent efforts in unconstrained handwritten page recognition are such as [Zimmerman and Bunke, 2002].

Writer identification has long been considered to be an important problem in handwriting recognition, since the variability of writings makes the problem of recognition difficult. Some adaptive solutions exist for the case of a mono-writer adaptation, including our own [Srihari and Bozinovic, 1985]. A process that would identify the writer, or style, could lead to a multi-writer approach that could be automatically adapted to the individual. The styles can be defined by characteristic elements of local patterns [Crettez, 1995]. Several invariant elements have been explored [Khan and Sethi, 1996], e.g., geometrical (loops, straight vertical lines), and topological (crossing points, extreme points). A novel approach has been the use of more global approaches based on fractals [Vincent, et. al., 2000].

2.2 Writer identification and Handwriting Matching

Writer identification has a long history and there exist many books describing the methodology employed by forensic document examiners, e.g., [Morris, 2000]. A computer system for retrieving the closest match from a large database of examples was developed by German law enforcement and is known as the FISH system [Filby, 2001].

Given handwriting exemplars whether a given set of measurements uniquely identifies a person is of great importance to justice and law enforcement systems. In the legal world numerous challenges have been made towards presenting expert forensic testimony in the courts regarding whether handwriting evidence has a scientific validation of its individuality. Long-established forensic handwriting examination has only recently faced this question and found that there is inadequate scientific support for the individuality argument. Such challenges, known as Daubert challenges have led to a need for a scientific demonstration of the individuality of handwriting.

Related to the issue of establishing the individuality is the need for associating a quantitative measure of similarity between two samples. Such a quantitative measure brings in an assurance of repeatability and hence a degree of objectivity. We have developed a theory and tested the theory with experiments for the task of handwriting identification as well as the specific subtask of handwriting verification (one where the task is to determine whether two documents were written by the same individual or by different individuals) [Srihari, et. al., 2002].

Handwriting matching is a two-class problem by measuring a “distance” between two samples and determining whether that distance could be classified as being within the same class or between different classes. It is necessary to choose an appropriate distance measure or measure of similarity in performing the transformation from feature-space to distance-space. If the features are continuous valued then the distances can be the differences in feature values. If the features are binary-valued then there are a number of methods of measuring similarity such as the Sokal-Michener measure, correlation, etc., [Zhang and Srihari, 2003].

3. FEATURE EXTRACTION

Both the tasks of handwriting recognition and writer identification need a set of features to be computed from the image. Both the tasks also involve classification: in recognition it is the task of classifying into the handwriting element: character or word, and in identification it is the task of classifying into the writer.

We compute two sets of features, known as macro- and micro-features, from the writing elements within the scanned and segmented handwritten images.

The macro-features, which represent the entire document, consist of three sets of features: *darkness*, *contour* and averaged *line-level* features. The darkness features, in turn, consist of three features all obtained from the histogram of the gray-scale values in the scanned document image: the number of black pixels in the image, the gray-scale value corresponding to the valley in the histogram that separates the foreground pixels from the background pixels (known as the threshold) and the entropy of the histogram (which is a measure of uncertainty in the distribution). The contour features, six in number, are as follows: the number of components and holes (as measured by the

number of interior and exterior contours in the chain-code outline of the handwriting), and slopes in the vertical, negative, positive and horizontal directions. The averaged line-level features consist of average slant and height of characters. The macro-features were normalized to lie in the [0,1] interval.

The micro-features are obtained by software previously developed for handwriting recognition [Srikantan, et. al., 1996]. The micro-features consist of 512 bits corresponding to *gradient* (192 bits), *structural* (192 bits), and *concavity* (128 bits) features. Each of these three sets of features relies on dividing the scanned image of the allograph (character or combination of characters) into a 4 x 4 region. The gradient features capture the frequency of the direction of the gradient, as obtained by convolving the image with a Sobel edge operator, in each of 12 directions and then thresholding the resultant values to yield a 192-bit vector. The structural features capture, in the gradient image, the presence of corners, diagonal lines, and vertical and horizontal lines, as determined by 12 rules. The concavity features capture, in the binary image, major topological and geometrical features including direction of bays, presence of holes, and large vertical and horizontal strokes.

The macro features are mapped into a distance vector of differences. The correlation similarity between the binary micro-feature vectors, are used to map the micro-feature vectors into distances.

In principle, the results of recognition of numerals and words can be used in the matching process as legibility features. They can also be used in the identification process to narrow down the set of writers from a large set of possible writers. The results of identification can be used in a bootstrapping process to retrieve a set of prototypes or features for that writer so that recognition can be performed more efficiently.

4. SYSTEM DESCRIPTION

We previously developed PENMAN, a system for handwriting recognition, at CEDAR [Srihari and Kim, 1997]. Elements of the PENMAN system, e.g., thresholding, line and word segmentation, are used to perform preprocessing operations in CEDARFOX.

The objective of CEDARFOX is to provide a measure of confidence of whether two samples (questioned document and exemplar document) belong to the same individual or to two different individuals.

CEDARFOX, which runs in a Windows environment, has a number of interactive features that makes it a useful as an experimental tool: image displays (Figure 2) feature extractors for handwriting, confidence value computation, etc. CEDARFOX also has preprocessors for handling scanned documents, such as underline removal, background thresholding, user specification of allographs (characters or combination of characters), etc. The user can also select as to which of the features need to be included in performing the comparison.

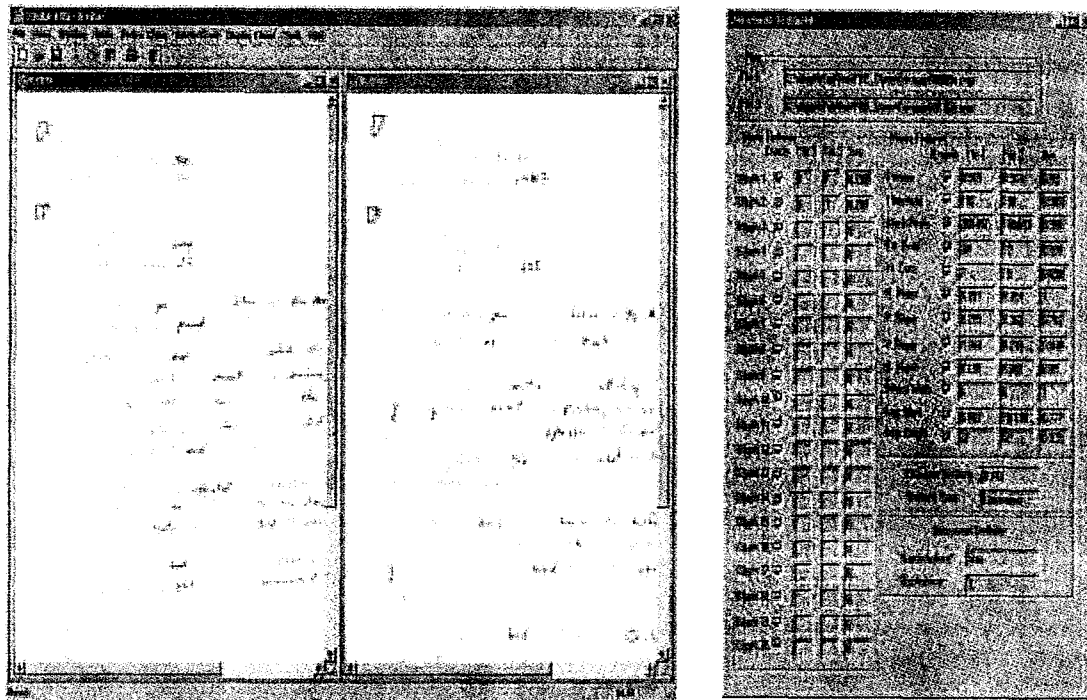


Fig. 2. Screen shots of CEDAR-FOX: (a) two handwritten pages for comparison, highlighted by automatic word segmentation performed by CEDAR-PENMAN, and (b) features and values which can be enabled/disabled by the user, together with a confidence of match.

The user can select which of the macro-features are used in performing the matching. The user can also select which of the characters in the two documents are to be used in making the comparison.

The capability of different handwritten characters to discriminate between individuals is shown in Figure 3. The discriminability measure is obtained by (i) obtaining the probability distributions of the distance between two samples conditioned on whether the samples belong to the same individual or to different individuals, and (ii) obtaining the ROC (receiver operating characteristics) curve for each character from the two distributions, and (iii) determining the area under each ROC curve for each character.

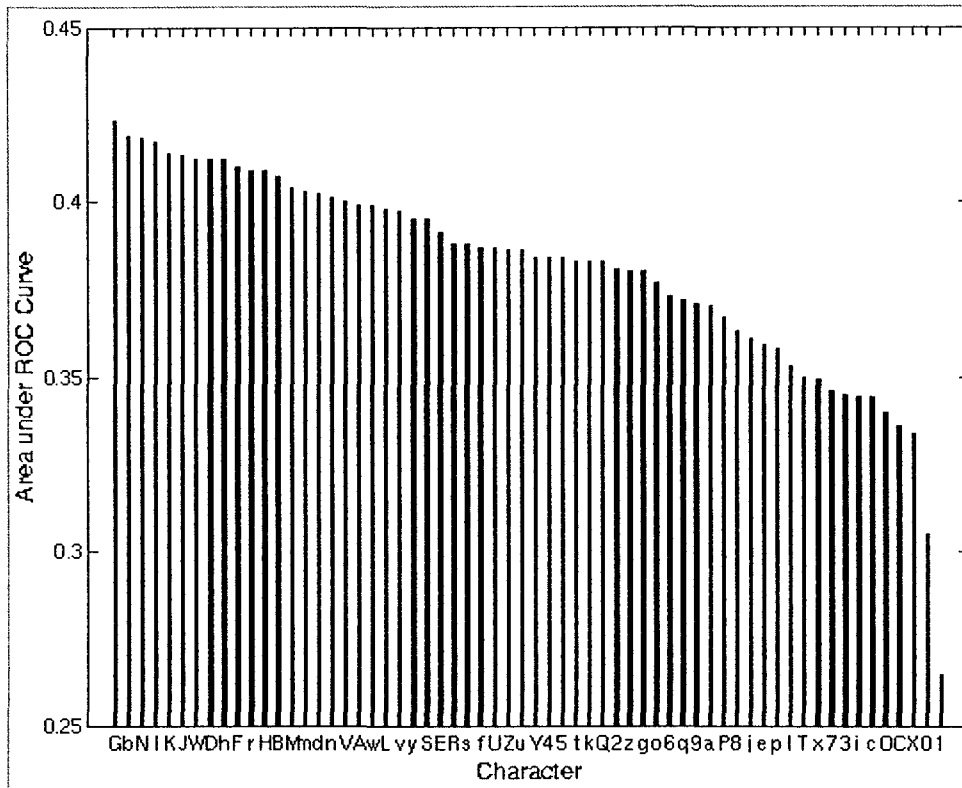


Fig.3. Discriminability of individuals based on characters: the letter G is most distinctive while the numeral 1 is least.

Discriminability is highest when all of the macro-features are used. It is least when only ten numerals (digits) in the handwritten document are used. Eleven alphabets, consisting of two small letters and nine capital letters, provide a higher degree of individuality than numerals.

While the present CEDAR-FOX is based on learning from the differences between exemplars of the same individual and between those of different individuals writing the same content, the planned system will learn from a large number of intra-writer samples of diverse content.

5. SUMMARY AND CONCLUSION

We have described an architecture and a system for computer processing of handwritten documents. It incorporates a system for handwriting matching based on learning from the differences between exemplars of the same individual and between those of different individuals. Handwriting recognition and writer identification are coupled by utilizing recognition results in identification and by using identification results to retrieve and employ writer characteristics in recognition. The CEDARFOX system for handwriting matching has a number of user-definable features and it is capable of producing confidence values in matching. Ongoing work at CEDAR will refine each of the functionalities described.

REFERENCES

- BOZINOVIC, R. M., and SRIHARI, S. N., 1987, "A multi-level perception approach to cursive Script recognition," *Artificial Intelligence Journal*, 33(2): 217-255.
- COHEN, E., HULL, J. J., and SRIHARI, S.N., 1994, "Control structures for interpreting handwritten addresses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1), pp. 66-75.
- CRETTEZ, J.-P., 1995, "A set of handwriting families: style recognition," *Proceedings of the Third International Conference on Document Analysis and Recognition (ICDAR)*, Montreal, Canada, pp. 489-494.
- HUBER, R.A., AND HEADRICK, A.M., 1999, *Handwriting Identification::Facts and Fundamentals*, Boca Roton: CRC Press.
- KHAN, K., and SETHI, I.K., 1996, "Handwritten signature retrieval and identification," *Pattern Recognition Letters*, 17: 83-90.
- LIU, C.L., SAKO, H., and FUJISAWA, H., 2002, "Integrated segmentation and recognition of handwritten numerals: comparison of classification algorithms," *Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition*, Niagara-on-the-lake, Canada, pp. 369-374.
- MAHADEVAN, U., and SRIHARI, S.N., 1999, "Parsing and recognition of city, state and ZIP Codes in handwritten addresses," *Proceedings of Fifth International Conference on Document Analysis and Recognition (ICDAR)*, Bangalore, India, pp. 325-328.
- NOSARY, A., HEUTTE, L., PAQUET, T., and LECOURTIER, Y., 1999, "Defining writer's invariants to adapt the recognition task," *Proceedings of the Fifth International Conference on Document Analysis and Recognition (ICDAR)*, Bangalore, India, pp. 765-768.
- PLAMONDON, R. and SRIHARI, S. N., 2000, "On-line and off-line handwriting recognition: A comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1): 63-84.
- SRIHARI, S.N., COHEN, E., HULL, J.J., and KUAN, L., 1989, "A system to locate and recognize ZIP Codes in handwritten addresses," *International Journal of Research and Engineering - Postal Applications*, 1(1): 37-56.
- SRIHARI, S.N., and KEUBERT, E.J., 1997, "Integration of handwritten address interpretation

- technology into the United States Postal Service Remote Computer Reader System," *Proceedings of the Fourth International Conference on Document Analysis and Recognition (ICDAR 97)*, Ulm, Germany, pp. 892-896 .
- SRIHARI, S.N., and KIM, G., 1997, "PENMAN: A system for reading unconstrained handwritten page images," *Proceedings of the Symposium on Document Image Understanding Technology (SDIUT 97)*, Annapolis, MD, pp. 142-153.
- SRIHARI, S. N., CHA, S-H., ARORA, H. AND LEE S., 2002, "Individuality of Handwriting", *Journal of Forensic Sciences*, 44(4): 856-72.
- SRIKANTAN. G, LAM S-W., AND SRIHARI, S. N., 1996, "Gradient-based contour encoding for character recognition," *Pattern Recognition*, 29: 1147-60.
- VINCENT, N., BOULETREAU, V., SABOURIN, R., and EMPTOZ, H., 2000, "How to use fractal dimensions to qualify writings and writers," *Fractals*, World Scientific, 8(1): 85-97.
- ZHANG, B., AND SRIHARI, S. N., 2003, "Binary vector dissimilarity measures for handwriting identification," T. Kanungo, Smith, E. H. B., Hu, J. and Kantor, P.B., eds., *Document Recognition and Retrieval X*, Bellingham, WA: SPIE, 5010: 28-38.
- ZIMMERMANN, M., and BUNKE, H., 2002, "Automatic segmentation of the IAM off-line Handwritten English databases, *Proceedings of Sixteenth International Conference on Pattern Recognition*, Quebec, Canada, pp. 35-39.
-

Indexing of Handwritten Historical Documents - Recent Progress

R. Manmatha Toni M. Rath
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts Amherst

Abstract

Indexing and searching collections of handwritten archival documents and manuscripts has always been a challenge because handwriting recognizers do not perform well on such noisy documents. Given a collection of documents written by a single author (or a few authors), one can apply a technique called word spotting. The approach is to cluster word images based on their visual appearance, after segmenting them from the documents. Annotation can then be performed for clusters rather than documents.

Given segmented pages, matching handwritten word images in historical documents is a great challenge due to the variations in handwriting and the noise in the images. We describe investigations into a number of different matching techniques for word images. These include shape context matching, SSD correlation, Euclidean Distance Mapping and dynamic time warping. Experimental results show that dynamic time warping works best and gives an average precision of around 70% on a test set of 2000 word images (from ten pages) from the George Washington corpus.

Dynamic time warping is relatively expensive and we will describe approaches to speeding up the computation so that the approach scales. Our immediate goal is to process a set of 100 page images with a longer term goal of processing all 6000 available pages.

1 Introduction

Libraries contain an enormous amount of handwritten historical documents. Such collections are interesting to a great range of people, be it for historians, students or just curious readers. Efficient access to such collections (e.g. on digital media or on the Internet) requires an index, for example like in the back of a book. Such indexes are usually created by manual transcription and automatic index generation from a digitized version. While this approach may be feasible for small numbers of documents, the

cost of this approach is prohibitive for large collections, such as the manuscripts of George Washington with well over 6000 pages.

Using Optical Character Recognition (OCR) as an automatic approach may seem like an obvious choice, since this technology has advanced enough to make commercial applications (e.g. tablet PCs) possible. However, OCR techniques have only been successful in the *online* domain, where the pen position and possibly other features are recorded during writing, and in *offline* applications (recognition from images) with very limited lexicons, such as automatic check processing (26 words allowed for legal amount field). However, for general historical documents with large lexicons, and the usually greatly degraded image quality (faded ink, ink bleed-through, smudges, etc.), traditional OCR techniques are not adequate. Figure 1 shows part of a page from the George Washington collection (this page is of relatively good quality).

Previous work [23] has shown the difficulties that even high-quality handwriting recognizers have with historical documents: the authors aligned a page from the Thomas Jefferson collection with a perfect transcription of the document. The transcription was used to generate a limited lexicon for each word hypothesis in the document. With a limited lexicon, the recognizer only has to decide which of a few possibilities the word to be recognized is. However, even for very small lexicon sizes of at most 11 (ASCII) words per word hypothesis in the image, only 83% of the words on a page could be correctly aligned with the transcription.

The wordspotting idea [12] has been proposed as an alternative to OCR solutions for building indexes of handwritten historical documents, which were produced by a single author: this ensures that identical words, which were written at different times, will have very similar visual appearances. This fact can be exploited by clustering words into groups with image matching techniques. Ideally, each clus-

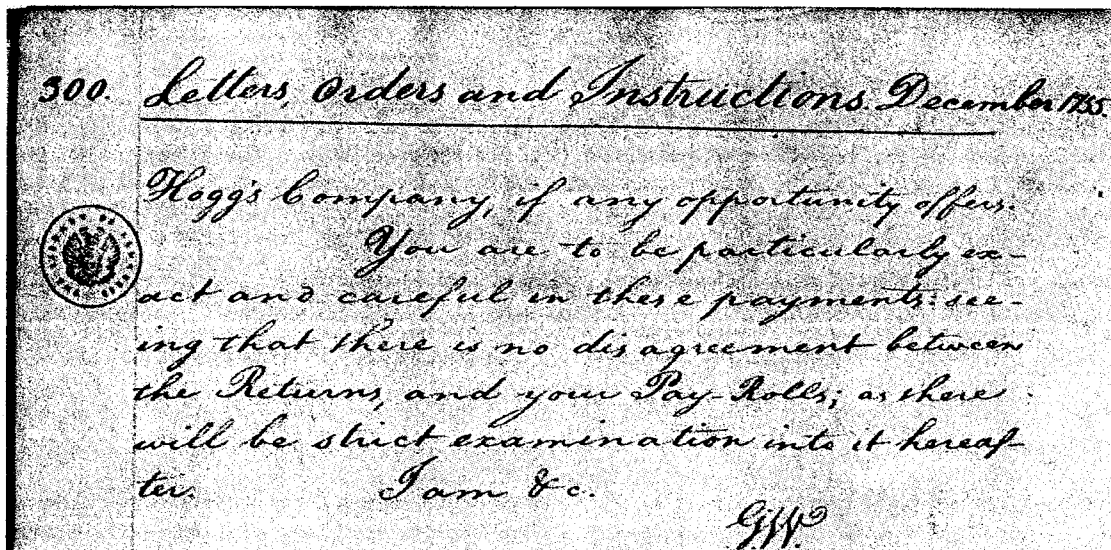


Figure 1: Part of a scanned document from the George Washington collection.

ter of word images consists of all occurrences of a word in the analyzed document collection. Clusters that contain the most words can then be annotated in order to allow an index generation for the respective words¹.

Apart from the problem of segmenting images from documents (see [14] for a scale-space approach), the crucial step is to determine the similarity between images of words. In this work, we present our recent investigations into robust matching techniques for words in historical manuscripts, and evaluate their performance on a subset of the George Washington collection.

2 Matching Techniques

The matching techniques we have investigated fall roughly into two categories: image matching approaches that compare images pixel-by-pixel, and feature-oriented techniques, that extract image features and compare them after determining correspondences between them.

2.1 Pixel-by-Pixel Matching

The matching techniques in this section compare two images pixel-by-pixel after aligning them initially. The alignments compensate part of the variations which are inherent to handwriting (e.g. shear and scale changes). Some of the matching techniques that have been investigated include:

1. XOR[8, 13]: The images are aligned and then a difference image is computed. The difference pixel count determines the cost.

¹Clusters of *stop* words, such as 'the' are not annotated.

2. SSD[8]: translates template and candidate image relative to each other to find the minimum cost (= matching cost) based on the Sum of Squared Differences.
3. EDM[8, 13]: Euclidean Distance Mapping. This technique is similar to XOR, but difference pixels in larger groups are penalized more heavily, because they are likely to result from structural differences between the template and the candidate image, not from noise.

Early versions of the above algorithms were proposed by [12, 13]. Kane et al. [8] improved them by using extensive normalization techniques that align images and also conducted more systematic experiments. The above algorithms (including the normalization techniques) are detailed in [8].

2.2 Feature-Oriented Matching

A number of feature based techniques have also been investigated for matching word:

1. SLH[8, 13]: recovers an affine warping transform (using the Scott and Longuet-Higgins algorithm [22]) between sample points taken from the edge of the template and candidate image. The residual between template points and warped candidate points is used as the matching cost.
2. SC [1, 15]: Shape Context matching. This algorithm is currently the best classifier for handwritten digits. Two shapes are matched by establishing correspondences between their outlines. The outlines are sampled and *shape context histograms* are generated for each sample

point: each histogram describes the distribution of sample points in the shape with respect to the sample point at which it is generated. Points with similar histograms are deemed correspondences and a warping transform between the two shapes is calculated and performed. The matching cost is determined from the cost associated with the chosen correspondences. [15] tested this algorithm for word matching in word spotting.

3. DTW[15]: A fixed number of features is extracted per image column, resulting in sets of “time series” (one per extracted feature) with the horizontal axis representing time. These time series can then be jointly aligned and compared with the Dynamic Time Warping algorithm. Examples of the time series features include projection profiles and upper/lower word profiles of the words.
4. CORR[18]: Using correlation, this technique recovers the correspondences between points of interest in two images. These correspondences are then used to construct a similarity measure.

Unlike the other techniques mentioned above, both the dynamic time warping and the point correspondence techniques share the property that they do not assume a global transformation between words². These two techniques turn out to be the best performing techniques with DTW being somewhat better than the other technique. In the following sections we will discuss the DTW and CORR techniques in some more detail. For more details of the use of the SLH and SC algorithm in word spotting, see [8] and [15].

2.3 Matching Words with DTW

A person writing a word usually moves the pen from left to right in English (from right to left in some languages like Arabic and Hebrew). While a word is inherently two dimensional, there is some association between image columns and the time they were written. By carefully pre-processing word images, one can minimize variations in the vertical dimension and then recast word matching as a 1-dimensional problem along the horizontal axis.

The slant and skew angles at which a person writes, are usually constant for single words, and can be normalized using a global transform. On the other hand, the inter-character and intra-character spacing is subject to larger variations. DTW [20] offers a way to compensate for these variations, which

²The warping transform used in the shape context algorithm is rather rigid.

is more flexible than linear scaling: in the matching algorithm that we describe here, image columns are aligned and compared using DTW. In our framework, each image column is represented by a time series sample point. Figure 2 shows an example alignment of two time series using dynamic time warping.

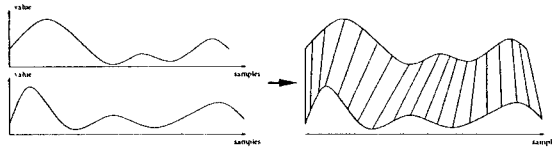


Figure 2: Alignment of two similar time series using dynamic time warping.

A single feature vector consists of one feature value per column of the image it is calculated for. For example, if image $A = (a(i, j))$ is w_A pixels wide, a feature $f(A)$ would be a vector of length w_A ³:

$$f(A) = (f(a(1, \cdot)), f(a(2, \cdot)), \dots, f(a(w_A, \cdot))). \quad (1)$$

The dynamic time warping matching algorithm simultaneously aligns two sets of feature vectors F_A and F_B which are extracted from the images A and B (F_B similarly):

$$F_A = (F_A(1, \cdot), F_A(2, \cdot), \dots, F_A(w_A, \cdot)), \quad (2)$$

where every entry $F_A(x, \cdot)$ is a d -dimensional vector consisting of all extracted feature values for image column x . That is, F_A and F_B consist of d individually calculated features that will be aligned together by the dynamic time warping algorithm. The matching error⁴ for matching images A and B is defined as

$$merr(A, B) = merr(F_A, F_B) = \frac{1}{l} DTW(w_A, w_B), \quad (3)$$

where l is the length of the warping path recovered by the dynamic time warping algorithm $DTW(\cdot, \cdot)$ which uses the recurrence equation

$$DTW(i, j) = \min \left\{ \begin{array}{l} DTW(i-1, j) \\ DTW(i, j) \\ DTW(i, j-1) \end{array} \right\} + d(i, j), \quad (4)$$

$$d(i, j) = \sum_{k=1}^d (F_A(i, k) - F_B(j, k))^2. \quad (5)$$

To prevent pathological warpings, global path constraints are used to force the paths to stay close to the diagonal of the DTW matrix [19]. More details of the dynamic time warping algorithm are presented in [15].

³The notation implies that every feature value is calculated strictly from the pixels in the corresponding image column. This constraint can be relaxed.

⁴Matching scores can be obtained from errors by negation.

Another approach, which uses dynamic time warping to compare features from a template word image to feature representations of whole lines of handwritten historical text, was described by Kolcz et al. in [10]. The main differences to our work are the application (retrieval by example in Kolcz’s case), the matching framework (independent alignment of time series vs. constrained alignment in our approach) and the limited evaluation (4 query examples vs. thousands in our case).

In the following section we present a number of features that we used for matching words as images, using the above dynamic time warping algorithm.

2.3.1 Features for Dynamic Time Warping

The images we operate on are all grayscale with 256 levels of intensity [0..255]. Before column features can be extracted from an image, a number of processing steps have to be performed: first, parts from other words that reach into this word’s bounding box have to be removed and the background is cleaned; then inter-word variations such as skew and slant angle have to be detected and normalized; next, the bounding box is cropped so that it tightly encloses the word; finally, the image is padded with extra rows either on top or on the bottom, to move the baseline⁵ to a predefined location. Figure 3 shows an original image and the result of the above processing steps.



(a) original image,



(b) cleaned and normalized version.

Figure 3: Original image and result after cleaning and normalization.

All of the column features we describe in the following are normalized to a maximum range of [0..1], so they are comparable across words. Our goal was to choose a variety of features presented in the handwriting recognition literature (e.g. [3] or [24]), such that an approximate reconstruction of a word from its features would be possible.

We use the following four features to represent word images (all of the example Figures were obtained from the image in Figure 3(b)):

⁵The baseline is the imaginary line people write on.

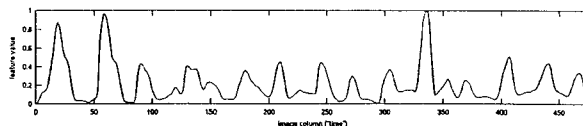


Figure 4: Projection profile feature (range-normalized and inverted).

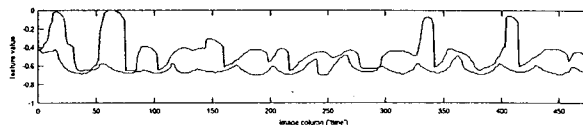


Figure 5: Upper and lower word profile features (normalized and inverted).

1. Projection Profile: these time series result from recording the sum of the pixel intensity values in every image column. The result is a profile that captures the distribution of ink along the horizontal axis of a word. Figure 4 shows a typical result.

2/3. Word Profiles: for every image, we can extract an upper and lower profile of the contained word, by going along the top (bottom) of the enclosing bounding box, and recording the distance to the nearest “ink” pixel in the current image column. Identifying ink pixels is currently realized by a thresholding technique, which we have found to be sufficient for our purposes. For more sophisticated foreground/background separation, see [11]. Together, these features capture the shape of the word outline (see Figure 5).

4. Background/Ink Transitions: for every image column, we record the number of transitions from a background- to an ink-pixel. This feature captures the inner structure of a word (see Figure 6 for an example).

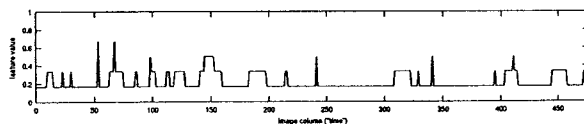


Figure 6: Background/ink transition-count feature (normalized).

We also tried a number of other features, including Gaussian derivatives and projection profiles that are calculated for parts of words. A discussion of the results can be found in [16].

2.3.2 Speeding up Dynamic Time Warping

While the word matching based on dynamic time warping works very well (see results in section 3), the computational load is quite high: for two images of width n , the algorithm's complexity is $O(n^2)$. Here we present preliminary investigations into possibilities for speeding up DTW.

Using a global path constraint like the Sakoe-Chiba band [19] speeds up the computation, because the DTW matrix only has to be evaluated in a region around the diagonal. The complexity of the resulting algorithm is still $O(n^2)$, but with a lower constant.

Another approach is to use the *lower-bounding* paradigm (e.g. see [5]): the idea is to use a lower-bounding function $lb(A, B)$, which always underestimates the real matching distance $merr(A, B)$:

$$\forall A \forall B : lb(A, B) \leq merr(A, B). \quad (6)$$

Such a lower-bounding function can be used for finding a time series in a collection, that has the lowest distance $merr$ to a given query series.

The approach is still to sequentially scan the data base for the best matching series, but lb is used to compare the query Q to a candidate C : if $lb(Q, C)$ is greater than the distance $merr(Q, M)$ to the currently best matching series M , it is not necessary to evaluate $merr(Q, C)$, since

$$merr(Q, M) < lb(Q, C) \leq merr(Q, C). \quad (7)$$

We only need to evaluate $merr(Q, C)$, if $lb(Q, C)$ is less than $merr(Q, M)$.

Of course, this strategy is only useful if lb has a lower complexity than $merr$. Several researchers have proposed lower-bounding functions for time series comparisons with DTW, with [9] being the tightest. The tightness is an important aspect of lower bounds, since it determines how often $merr$ has to be evaluated for time series that do not yield a distance which is lower than the current minimum. The lower bound in [9] was proposed for univariate time series. We have extended the approach to multivariate time series [17].

2.4 Word Matching using Point Correspondences

The image matching approach based on point correspondences identifies image corners in the input images using the Harris detector. Then, the similarity between these points is determined by correlating their intensity neighborhoods using the sum of squared differences measure. Then the recovered correspondences are used to calculate a measure of similarity between the input word images.

Recovering correspondences for all pixels in one word image would be an expensive operation, considering the search space, which is of quadratic size in the number of sample points in the two input images. Using the Harris detector allows us to select a limited number of points of interest, which are repeatable under a range of transformations and invariant to illumination variations [6].

The Harris detector operates on the matrix

$$M = \begin{pmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \left(\frac{\partial I}{\partial y}\right)^2 \end{pmatrix},$$

where I is the gray level intensity image. A corner is determined when the 2 eigenvalues of M are large since this indicates grayscale variations in both the x and y direction.

2.4.1 Recovering Corner Correspondences

Here we determine pairs of corresponding corner points in the two input images. Most correspondence methods compare the characteristics of the local regions around feature points, and then select the most similar pairs as correspondences. The characteristics of local regions can be represented by either a feature vector (e.g. see [21]), or by windows of gray-level intensities.

We use the sum of squared differences (SSD) error measure to compare gray-level intensity windows, which are centered around detected corner locations. The reasons for selecting the SSD measure are its simplicity and the small number of operations required to calculate it - an important consideration when comparing a large number of image pairs.

In this simple approach, false point matches can be caused by a number of factors. We try to alleviate them with constraints:

- the size of the query word may be different from that of the candidate word image, that is, they have different resolutions. Assuming tight bounding boxes for all words, we resize all candidate images to the size of the query image.
- For a given feature point, there might be several feature points in a candidate image, which result in small SSD errors. To reduce this possibility, we constrain corresponding feature points in the candidate image to lie in the neighborhood of the corner point in the template image. This constraint also has the desirable effect of speeding up the algorithm.

In order to further decrease the computational load, we reduce each image to half-size. In essence, this can be regarded as doubling the size of the SSD

correlation windows without slowing down the implementation. Using larger SSD windows can help prevent false matches, because of the added context that is taken into account when comparing image regions.

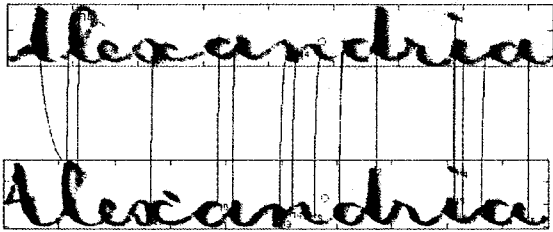


Figure 7: Recovered correspondences in two word images.

Experiments showed that adding the above constraints greatly improved the matching accuracy (see Figure 7 for an example of recovered correspondences).

2.4.2 Distance Measure Calculation

The correspondence between pairs of feature points captures the similarity between *local* regions of two images. In order to judge the similarity of two word images, this local information is now combined into a *global* measure of similarity.

After investigating various approaches for distance measurements, we used the following distance measurement:

$$D(A, B) = \frac{\sum_i \sqrt{(x_{bi} - x_{ai})^2 + (y_{bi} - y_{ai})^2}}{\# \text{correspondences}} \cdot \frac{\# \text{feature points in A}}{\# \text{correspondences}}, \quad (8)$$

where A is the query image, and B a candidate image; (x_{ai}, y_{ai}) and (x_{bi}, y_{bi}) are the coordinates of a pair of corresponding feature points, in A and B respectively. Essentially, we are calculating the mean Euclidean distance of corresponding feature points. Additionally, we penalize for every point in image A, that does not have a correspondence in image B⁶ by multiplying the average distance with a weight. Thus, the fewer corresponding feature points are found in the candidate image B, the larger the distance between B and A.

More details on the point correspondence technique can be found in [18].

3 Results

We conducted experiments on two labeled data sets, both 10 pages in size. Data set 1 is of acceptable

⁶This can happen if the search area in B for a corner point in A does not contain any points of interest.

quality (see Figure 8(a)). The second set is very degraded (see Figure 8(b)) - even humans have difficulties reading these pages. We prepared four test sets for our evaluation:

- A: 15 images in test set 1.
- B: 2372 images of good quality (test set 1).
- C: 32 images in test set 2.
- D: 3262 images of bad quality (test set 2).

Test sets A and C are mainly for the purpose of comparison to previously reported techniques (in [8]) and for quick performance tests of new techniques.

3.1 Experimental Methodology

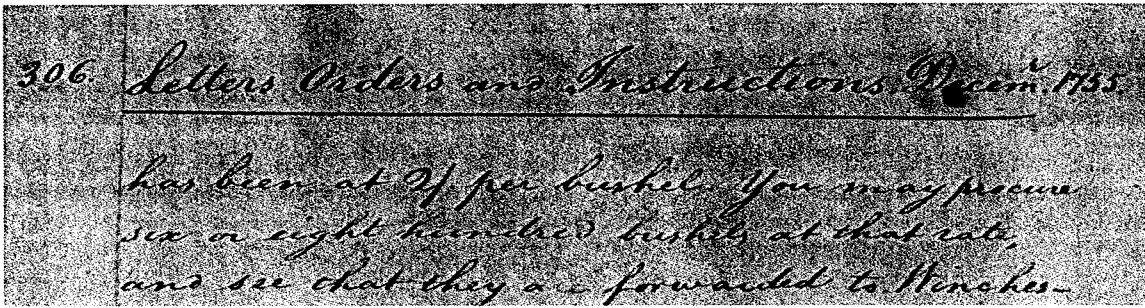
For a given test set/matching algorithm pair, the following evaluation was performed: each image in the test set was regarded as a query, which was used to rank the rest of the images in the collection according to their similarity to the query. Some query/candidate pairs are not compared by the matching function (“pruned”), because they are dissimilar according to a set of simple heuristics (image length, aspect ratio of bounding box, etc.).

The ranked lists of retrieved word images were evaluated using the mean average precision measure [25], which is commonly used in the information retrieval field. For the purpose of evaluation, a candidate image was considered “relevant” to the query image, if the image labels (ASCII annotation) matched.

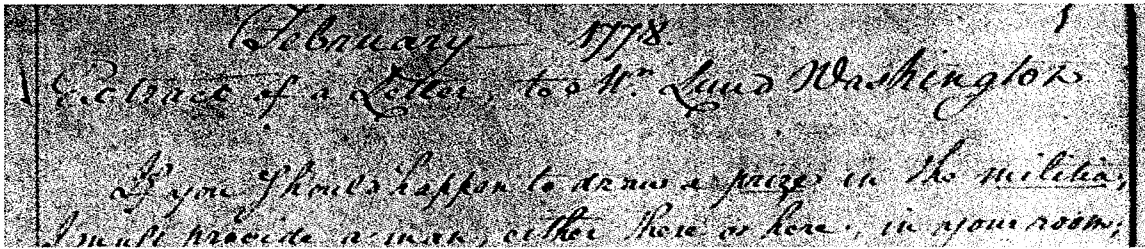
Previous work by [8] used a somewhat misleading evaluation, that considered each query image as a candidate. With this evaluation, the results are biased, since most of the techniques always retrieve the query image at rank 1, if it is part of the candidate set. In table 1, we have provided both the old evaluation and a new version, which removes query images from the candidate set. The new evaluation values for some of the discussed matching techniques appear in the 4 right-most columns.

3.2 Result Discussion

As can be seen from table 1, DTW and CORR work best, with similar performance on all data sets. While CORR performs better on the smaller sets A and C, DTW seems to have a slightly better overall performance (data sets B and D). The EDM matching technique seems to perform well on data set A, but its performance is significantly lower on set C. Additionally, it is unclear what its overall performance is, since we had no raw results available for sets B and D. The rest of the matching approaches (XOR, SSD, SLH and SC) does not perform nearly



(a) example from test set 1 (good quality),



(b) example from test set 2 (bad quality).

Figure 8: Examples from the two test sets used in the evaluation.

| Run | XOR | SSD | SLH | SC | EDM | DTW | CORR | SC | EDM | DTW | CORR |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| A | 54.14% | 52.66% | 42.43% | 48.67% | 72.61% | 73.71% | 73.95% | 40.58% | 67.67% | 67.92% | 69.69% |
| B | n/a | n/a | n/a | n/a | n/a | 65.34% | 62.57% | n/a | n/a | 40.98% | 36.23% |
| C | n/a | n/a | n/a | n/a | 15.05% | 58.81% | 59.96% | n/a | n/a | 13.04% | 14.84% |
| D | n/a | n/a | n/a | n/a | n/a | 51.81% | 51.08% | n/a | n/a | 16.50% | 15.49% |

Table 1: Average precision scores for all test runs (XOR: matching using difference images, SSD: sum of squared differences technique, SLH: technique by Scott & Longuet-Higgins [22], SC: shape context matching [1], EDM: euclidean distance mapping, DTW: dynamic time warping matching, CORR: recovered correspondences). Four right-most columns show corrected evaluation results.

as well as DTW and CORR, with mean average precision values in the 40-50% range.

The general performance difference of DTW and CORR on data sets A and B can be explained by the pruning heuristics, which work much better on set A than on set B: in set A, only 10% of the valid matches are discarded in the pruning, while in set B, almost 30% are discarded. A similar observation can be made for the test sets C and D: on both sets, the pruning discards around 45% of the valid matches. This can be seen in the smaller differences in mean average precision for both DTW and CORR on these data sets. The reason for the high rejection rate of valid matches by the pruning lies in the word segmentation, which is heavily affected by the bad quality of the document images.

4 Conclusions and Outlook

Given the challenges in recognizing words from the large vocabularies of handwritten manuscript collections, word spotting involving word matching is a reasonable approach for solving the problem of in-

dexing such manuscript collections. We have discussed a number of different approaches to matching with the best performing ones being dynamic time warping and a point correspondence based technique. Challenges remain, including the creation of word clusters and the necessity of speeding up these algorithms sufficiently, so that large collections can be handled in a reasonable amount of time.

Building a system involves creating a user interface. While it is straightforward to imagine a visual index with pictures and links to pages, it is not clear whether users would be able to use such an index effectively. An ASCII user interface can be created by annotating the matched word clusters manually - permitting a more traditional index. Recent advances in automatic picture annotation [2, 4, 7], using machine learning and information retrieval techniques, may permit a completely different approach to this problem through automatic annotation of word image clusters.

Acknowledgments

We would like to thank Jamie Rothfeder and Shaolei Feng for contributing to the work on point correspondences for word spotting. We also thank the Library of Congress for providing the images of the George Washington collection.

This work was supported in part by the Center for Intelligent Information Retrieval and in part by the National Science Foundation under grant number IIS-9909073. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

References

- [1] S. Belongie, J. Malik and J. Puzicha: *Shape Matching and Object Recognition Using Shape Contexts*. IEEE Trans. on Pattern Analysis and Machine Intelligence **24**:24 (2002) 509-522.
- [2] D. M. Blei and M. I. Jordan: *Modeling Annotated Data*. Technical Report UCB//CSD-02-1202, 2002.
- [3] C.-H. Chen: *Lexicon-Driven Word Recognition*. In: Proc. of the Third Int'l Conf. on Document Analysis and Recognition 1995, Montréal, Canada, August 14-16, 1995, pp. 919-922.
- [4] P. Duygulu, K. Barnard, N. de Freitas and D. Forsyth: *Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary*. In: Proc. 7th European Conference on Computer Vision, Copenhagen, Denmark, May 27-June 2, 2002, vol. 4, pp. 97-112.
- [5] C. Faloutsos: *Multimedia IR: Indexing and Searching*. In: Modern Information Retrieval, R. Baeza-Yates and B. Ribeiro-Neto; Addison-Wesley, Reading, MA, 1999.
- [6] C. Harris and M. Stephens: *A Combined Corner and Edge Detector*. In: Proc. of the 4th Alvey Vision Conf., 1988, pp. 147-151.
- [7] J. Jeon, V. Lavrenko and R. Manmatha: *Automatic Image Annotation and Retrieval Using Cross-Media Relevance Models*. CIIR Technical Report MM-41, 2003.
- [8] S. Kane, A. Lehman and E. Partridge: *Indexing George Washington's Handwritten Manuscripts*. Technical Report MM-34, Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2001.
- [9] E. Keogh: *Exact Indexing of Dynamic Time Warping*. In: Proc. of the 28th Very Large Databases Conf. (VLDB), Hong Kong, China, August 20-23, 2002, pp. 406-417.
- [10] A. Kolcz, J. Alspector, M. Augusteijn, R. Carlson and G. V. Popescu: *A Line-Oriented Approach to Word Spotting in Handwritten Documents*. Pattern Analysis & Applications **3** (2000) 153-168.
- [11] G. Leedham, S. Varma, A. Patankar and V. Govindaraju: *Separating Text and Background in Degraded Documents Images - A Comparison of Global Thresholding Techniques for Multi-Stage Thresholding*. In: Proc. of the 8th Int'l Workshop on Frontiers in Handwriting Recognition 2002, Niagara-on-the-Lake, ON, August 6-8, 2002, pp. 244-249.
- [12] R. Manmatha, C. Han, E. M. Riseman and W. B. Croft: *Indexing Handwriting Using Word Matching*. In: Digital Libraries '96: 1st ACM Int'l Conf. on Digital Libraries, Bethesda, MD, March 20-23, 1996, pp. 151-159.
- [13] R. Manmatha and W. B. Croft: *Word Spotting: Indexing Handwritten Archives*. In: Intelligent Multi-media Information Retrieval Collection, M. Maybury (ed.), AAAI/MIT Press 1997.
- [14] R. Manmatha and N. Srimal: *Scale Space Technique for Word Segmentation in Handwritten Manuscripts*. In: Proc. 2nd Int'l Conf. on Scale-Space Theories in Computer Vision, Corfu, Greece, September 26-27, 1999, pp. 22-33.
- [15] T. M. Rath and R. Manmatha: *Word Image Matching Using Dynamic Time Warping*. to appear in Proc. of the Computer Vision and Pattern Recognition Conf. 2003.
- [16] T. M. Rath and R. Manmatha: *Features for Word Spotting in Historical Manuscripts*. to appear in Proc. of the 7th Int'l Conf. on Document Analysis and Recognition 2003.
- [17] T. M. Rath and R. Manmatha: *Lower-Bounding of Dynamic Time Warping Distances for Multivariate Time Series*. CIIR Technical Report MM-40, 2003.
- [18] J. L. Rothfeder, S. Feng and T. M. Rath: *Using Corner Feature Correspondences to Rank Word Images by Similarity*. CIIR Technical Report MM-44, 2003.
- [19] H. Sakoe and S. Chiba: *Dynamic Programming Optimization for Spoken Word Recognition*. IEEE Trans. on Acoustics, Speech and Signal Processing **26** (1980) 623-625.

- [20] D. Sankoff and J. B. Kruskal: *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, Reading, MA, 1983.
- [21] C. Schmid and R. Mohr: *Local Grayvalue Invariants for Image Retrieval*. IEEE Trans. on Pattern Analysis and Machine Intelligence **19**:5 (1997) 530-535.
- [22] G. L. Scott and H. C. Longuet-Higgins: *An Algorithm for Associating the Features of Two Patterns*. Proc. of the Royal Society of London **B224** (1991) 21-26.
- [23] C. I. Tomai, B. Zhang and V. Govindaraju: *Transcript Mapping for Historic Handwritten Document Images*. In: Proc. of the 8th Int'l Workshop on Frontiers in Handwriting Recognition 2002, Niagara-on-the-Lake, ON, August 6-8, 2002, pp. 413-418.
- [24] Ø. D. Trier, A. K. Jain and T. Taxt: *Feature Extraction Methods for Character Recognition - A Survey*. Pattern Recognition **29**:4 (1996) 641-662.
- [25] C. J. van Rijsbergen: *Information Retrieval*. Butterworth, London, England, 1979.

Document Categorization Using Latent Semantic Indexing

Anthony Zukas Robert J. Price

Science Applications International Corporation
1953 Gallows Road, Vienna, VA 22182, U.S.A.
Anthony.E.Zukas@saic.com

Abstract

The purpose of this research is to develop systems that can reliably categorize documents using the Latent Semantic Indexing (LSI) technology [2]. Initial research has indicated that the LSI technology shows great promise in constructing categorization systems that require minimal setup and training. Categorization systems based on the LSI technology do not rely on auxiliary structures (thesauri, dictionaries, etc.) and are independent of the native language being categorized (given the documents can be represented in the UNICODE character set).

Three factors led us to undertake an assessment of LSI for categorization applications. First, LSI has been shown to provide superior performance to other information retrieval techniques in a number of controlled tests [3]. Second, a number of experiments have demonstrated a remarkable similarity between LSI and the fundamental aspects of the human processing of language [6]. Third, LSI is immune to the nuances of the language being categorized, thereby facilitating the rapid construction of multilingual categorization systems.

The emergence of the World Wide Web has led to a tremendous growth in the volume of text documents available to the open source community (e.g., special interest web pages, digital libraries, subscription news sources, and company-wide Intranets). Quite coincidentally, this has led to an equally explosive interest in accurate methods to filter, categorize and retrieve information relevant to the end consumer. Of special emphasis in such systems is the need to reduce the burden on the end consumer and minimize the system administration of the system.

We will describe the implementation of two successfully deployed systems employing the LSI technology for information filtering (English and

Spanish language documents) and document categorization (Arabic language documents). The systems utilize in-house developed tools for constructing and publishing LSI categorization spaces. Various interfaces (e.g., SOAP-based Web service, workflow interfaces, etc.) have been developed that allow the LSI categorization capability to address a variety of customer system configurations. The core LSI technology has been implemented in a modern J2EE based architecture facilitating its deployment on a variety of platforms and operating systems. We will describe some early results on the accuracy and use of the systems.

Introduction

Latent Semantic Indexing is an automated technique for the processing of textual material. It provides state-of-the-art capabilities for:

- automatic document categorization;
- conceptual information retrieval, and;
- cross-lingual information retrieval.

A key feature of LSI is that it is capable of automatically extracting the conceptual content of text items. With knowledge of their content, these items then can be treated in an intelligent manner. For example, documents can be routed to individuals based on their job responsibilities. Similarly, e-mails can be filtered accurately. Information retrieval operations can be carried out based on the conceptual content of documents, not on the specific words that they contain. This is very useful when dealing with technical documents, particularly cross-disciplinary material.

LSI is not restricted to working with words; it can process arbitrary character strings. For example, tests with MEDLINE data have shown that it deals effectively with chemical names. Points in an LSI space can represent any object that can be expressed in terms of text. LSI has been used with great success in representing user interests and the expertise of individuals. As a

result, it has been employed in applications as diverse as capturing customer preferences and assigning reviewers at technical conferences.

In cross-lingual applications, training documents from one language can be used to categorize documents in another language (for languages where a suitable parallel corpus exists). A discussion on LSI's cross-lingual capabilities can be found in [4].

Training

Text categorization is the assignment of natural language texts to one or more predefined categories based on their content [1]. Text categorization systems run the gamut from those that employ trained professionals to categorize new items to those that are based on natural language clustering algorithms requiring no human intervention to guide the categorization process; the former process being very time consuming and costly, while the latter is the pinnacle of text categorization. Practical, cost-effective implementations fall somewhere in between.

Supervised text categorization has a learning (or training) component where pre-defined category labels are manually assigned to a set of documents that will become the basis for subsequent automated categorization. Text categorization systems performing unsupervised training (or learning) automatically detect clusters or other common themes in the data that identify topics or labels without manual labeling of the data.

When used in text categorization applications LSI requires a labeled training set of documents. Labeled training sets can be as few as seventy-five to several thousand in number. It is possible to use a small number of labeled documents to bootstrap the supervised learning process. After building an initial index with labeled test documents, additional documents can be submitted as queries, and query documents close in similarity to labeled documents in the index (within some pre-specified threshold value) can then be associated with the same label. In this manner the labeled test set can be grown over time with a significant reduction in the human effort required to build a large labeled test set.

When using smaller-sized training sets (less than 300-600 documents) LSI may require some

additional tuning of the dimensionality of the categorization index to capture the higher ranked latent features in the training set. This is easily accomplished through a graphical user interface and iterations through re-indexing of the training set.

We have also found that adding unlabeled data ("background" text) in the presence of small labeled test sets improves the latent structure of the categorization index leading to improved accuracy. Similar results have been reported in the literature [7, 10]. Figure 1 shows the effect of background material on a small labeled test set of 300 documents. Unlabeled examples (e.g., web pages, emails, news stories) are much easier to locate and collect than labeled examples.

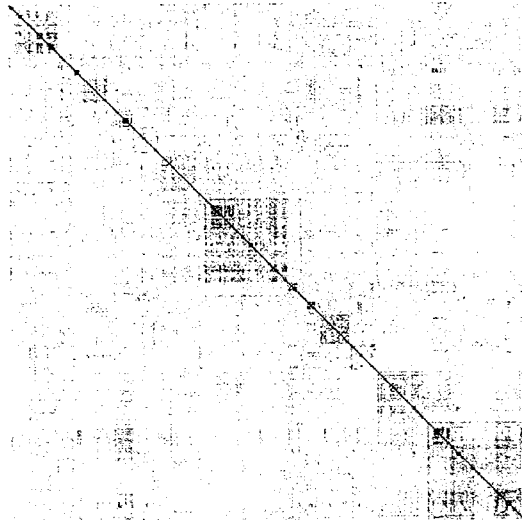
A common critique of LSI in the literature is the relatively high computational and memory requirements required by LSI to function [5]. However, with the advent of modern processors and their ever-increasing speeds this former consideration has been overcome. Training LSI with a moderate training set can be accomplished in a matter of minutes on current corporate desktop PCs with less than 1GB of memory. Larger sets of training documents require less than 10 minutes on equivalent PCs.

Test Corpus and Performance Measures

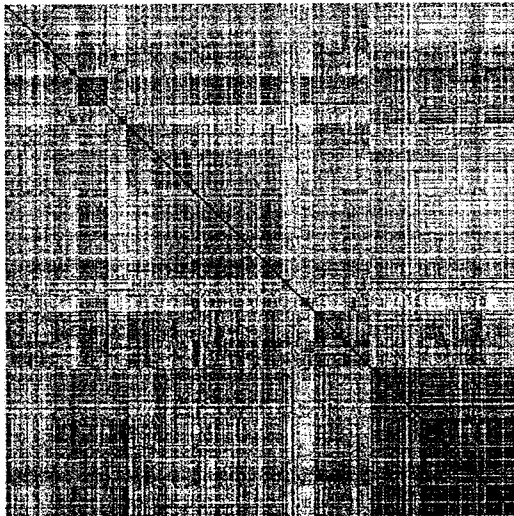
LSI as a text categorization engine has been deployed in a number of real world applications as described below. To compare its performance to other published results we used the ModApte version of the Reuters-21578 test set [11]. The ModApte version has been used in a wide number of studies [8] due to the fact that unlabeled documents have been eliminated and categories have at least one document in the training set and the test set. We followed the ModApte split defined in the Reuters-21578 data set in which 71% of the articles (6552 articles) are used as labeled training documents and 29% of the articles (2581 articles) are used to test the accuracy of category assignments.

Many different evaluation criteria have been used for evaluating the performance of categorization systems. For evaluating the effectiveness of category assignments to documents by LSI we adopted the *break-even point* (the arithmetic average of precision and

recall) as reported in [1] and [8], and the total ('micro-averaged') precision P and recall R (defined in [9]). The micro-averaged break-even point is defined as $(P+R)/2$.



(a)



(b)

Figure 1. Graphic of the training space illustrating the effects of background material on the LSI training set. Figure 1a shows the similarity matrix for the training set. The row and column axes represent the documents in the training set; the diagonal shows that every document is related to itself. The stronger outlines surrounding the diagonal represent the labeled classes within the training data. Figure 1b shows how background material strengthens the latent relationships in the training data.

Comparison with Other Techniques

Table 1 summarizes the global performance score for LSI along with the best performing classifier from [8]. As can be seen from Table 1, the LSI miF1 value was competitive with the miF1 value for the Support Vector Machine (SVM) in [8].

Table 1. Performance summary

| Method | miR | miP | miF1 | MaF1 | Error |
|------------|-------|-------|-------|-------|--------|
| LSI | .8880 | .8900 | .8890 | .5880 | .004 |
| SVM [8] | .8120 | .9137 | .8599 | .5251 | .00365 |

miR=micro-avg recall; miP=micro-avg prec.;
miF1=micro-avg F1; maF1=macro-avg F1.

While the document counts between the two studies were not exactly the same the overall ratios of training set to test set were almost exactly the same; in [8] the ratios were 72 and 28 percent for the training and test sets, respectively. Additionally, in [8] there was an assumption that documents could fit into more than one category; unlabeled documents were eliminated and categories had to have at least one document in a training set and the test set. In [8] the number of categories per document was 1.3, on average. The category per document ratio for the ModApte data set used in this paper was one. This is a more stringent restriction on text categorization classifiers. The LSI results reported in Table 1 reflect this constraint.

In [1] the assumptions concerning what documents made up the ModApte split differ slightly from [8] and the test set used in this study. The mean number of categories per document for [1] was 1.2, but many documents were not assigned to any of the 118 categories, and some documents were assigned to 3 or more categories. The Support Vector Machine (SVM) was the most accurate text categorization method in [1] with an overall miF1 rating of 0.8700 placing it between the LSI and SVM results reported in Table 1.

Real World Applications

Information Filtering/Knowledge Discovery

In this application, the customer had a proprietary process for collecting English and Spanish content on a periodic basis. Once

collected, the content was indexed with Boolean retrieval technology and made available to analysts for review. Analysts constructed and executed queries to retrieve content specific to their particular interests. Results varied depending on the expertise analysts possessed in constructing queries. An additional drawback was that analysts spent a large amount of their time searching for relevant content rather than analyzing content.

To address the above situation, LSI technology was integrated into the workflow to replace the Boolean retrieval technology. Rather than construct and execute queries analysts supplied representative content (i.e., documents) relevant to their areas of interest. This material was tagged, and indexed. Content collected on a periodic basis was compared to the index of analyst relevant content. Content similar in nature (within a specified threshold) to analyst content was routed to the appropriate analyst. Restructuring of the workflow in this manner resulted in a continuous push of relevant content to analysts, resulting in a significant increase in productivity on the part of the analyst. This system has been in production for over two years.

Document Categorization/Prioritization

In this application the customer had a high volume of Arabic language content and an insufficient number of Arabic-qualified analysts to review all the content. In order to ensure that relevant content was not overlooked all of the material had to be examined leading to overworked analysts and a situation where, potentially, some item of important material might be overlooked.

To address the above situation, a training set of Arabic content was constructed and labeled according to customer-defined categories. The system was trained with the labeled training set. An additional 20,000 relevant Arabic documents were selected and used as background training material. Integration with the customer's workflow was accomplished using a SOAP-based web service. Arabic documents for categorization are passed to the web service. A ranked list of categories and associated similarity scores are sent back to the client process. Based on customer-defined rule sets, the client process makes decisions about the importance of the documents and their disposition. Highly ranked documents are immediately forwarded to

analysts, less important documents are stored for later examination during periods of analyst workload, and uninteresting documents were discarded. This system has been in production for one year. During customer acceptance testing, this system demonstrated 97% accurate assignment of Arabic-language documents to individual categories. This result was measured using real-world documents with significant quantities of noise.

Conclusions

The LSI technology has matured to the point where it is a particularly attractive approach for text categorization. Text categorization results with LSI are competitive, on comparable test sets, with the best results reported in the literature. A definite advantage to the LSI text categorization technology is the native support for international languages.

Lessons Learned

LSI categorization can perform well with very limited quantities of training data – generally with only a few examples per category. This is due, in great part, to the exceptional conceptual generalization capabilities of LSI.

User feedback can be incorporated to continually improve performance.

The LSI technique has a significant degree of inherent noise immunity with regard to errors in the documents being processed.

Documents can be assigned to multiple categories, with reliable indications of the degree of similarity to each category.

Acknowledgements

We thank Roger Bradford, Janusz Wnek, and Rudy Keiser for their useful comments when reviewing this paper.

References

- [1] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. *Inductive learning algorithms and representations for text categorization*, Proceedings of ACM-CIKM'98, 1998.
- [2] S. Deerwester et al. *Indexing by Latent Semantic Analysis*, *Journal of the Society for Information Science*, 41(6), pp. 391-407, October, 1990.

- [3] S. Dumais. *Using LSI for Information Retrieval, Information Filtering, and Other Things*, Cognitive Technology Workshop, April 4-5, 1997.
- [4] S. Dumais et al. *Automatic Cross-linguistic Information Retrieval using Latent Semantic Indexing*, in SIGIR'96 - Workshop on Cross-Linguistic Information Retrieval, pp. 16-23, August 1996.
- [5] G. Karypis and E. Han. *Fast supervised dimensionality reduction algorithm with applications to document categorization and retrieval*, Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management, 2000.
- [6] T. Landauer and D. Lanham. *Learning Human-like Knowledge by Singular Value Decomposition: a Progress Report*, Advances in Neural Information Processing Systems 10, Cambridge: MIT Press, pp. 45-51, 1998.
- [7] K. Nigam. *Using unlabeled data to improve text classification*, PhD. Thesis, Carnegie Mellon University, May 2001.
- [8] Y. Yang and X. Liu. *A re-examination of text categorization methods*, Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99), pp. 42-49, 1999.
- [9] Y. Yang. *An evaluation of statistical approaches to text categorization*, Journal of Information Retrieval, Volume 1, No. 1/2, pp. 67-88, 1999.
- [10] S. Zelikovitz and H. Hirsh. *Using LSI for Text Classification in the Presence of Background Text*, Proceeding of CIKM-01, 10th ACM International Conference on Information and Knowledge Management, 2001.
- [11] The Reuters-21578 collection is available at:
<http://www.davidlewis.com/resources/testcollection/reuters21578/>

Parsing Freeform Handwritten Notes on the Tablet PC

Michael Shilman

(joint work with Zile Wei, Sashi Raghupathy, Patrice Simard, David Jones)
Microsoft, Inc.

ABSTRACT

The Tablet PC is a great platform for taking handwritten notes in digital ink, a medium that is flexible to capture text, diagrams, tables, and annotations at the speed of a discussion. However, once the notes have been captured recognizing the text in those notes is difficult. Text-graphics classification and text layout analysis are classical problems in printed document analysis, but the irregularity in handwriting and content in freeform notes reveal limitations in existing approaches. We advocate an integrated approach that looks at the layout and classification problems simultaneously. We evaluate our approach both qualitatively and with an accuracy metric based on tree edit distance, and reflect on some of the difficult recognition scenarios that we have encountered.

Degraded Documents

Processing Noisy Documents

Yefeng Zheng, Huiping Li and David Doermann
Language and Media Processing Laboratory
Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742-3275
E-mail: {zhengyf, huiping, doermann}@cfar.umd.edu

Abstract

In this paper we present an overview of our work on processing noisy documents, including the identification of handwriting, and the detection and removal of background lines. To identify handwriting, we treat noise as a distinguished class and model noise based on selected features. We then use a trained Fisher classifier to separate machine printed text and handwriting from noise. A Markov Random Field (MRF) based approach is applied to incorporate context and refine the classification. To detect and remove background lines which may touch the text and handwriting, we propose a model based approach which incorporates high level contextual information to detect severely broken parallel lines. We use a Directional Single-Connected Chain(DSCC) method to extract the line segments, and then construct a parallel line model with three parameters: the skew angle, the vertical line gap, and the vertical translation. A coarse-to-fine approach is used to improve the estimate. Our experimental results show our method can detect 94% of the lines in a database of 168 noisy Arabic document images.

1 Introduction

Automatic document processing and analysis has advanced to a point that text segmentation and recognition are viewed as solved problems for clean, well-constrained documents. However, the performance degrades quickly even when a small amount of noise is introduced. For example, a typical bottom-up page segmentation method starts from the extraction of connected components [1], which are then merged into words, text lines, and zones. After segmentation, a classification module is used to identify zone types, such as text, table and image. These algorithms work well on clean documents where zones with different properties can easily be separated. However, they fail on noisy documents, where noise often mixes with and/or is spatially close to text regions.

In this paper we give an overview of our work on processing noisy documents, including the identification of handwriting, and background line detection and removal. Handwritten annotations often indicate corrections, additions or other supplemental information that should be treated differently from the main or body content. We have found annotations of particular interest in the processing of correspondence and related business documents. Since it is not uncommon that background lines exist in the documents, touching or mixing with handwriting and text (Figure 3a), it is important that those lines be detected and removed before we feed the text to an Optical Character Recognition (OCR) engine.

Table 1. Features used for classification.

| | Usage description | Dimension | Selected |
|---------------------------|--------------------------------------|-----------|----------|
| Structural | Region size, connected components | 18 | 9 |
| Gabor filter | Stroke orientation | 16 | 4 |
| Run-length histogram | Stroke length | 20 | 5 |
| Crossing counts histogram | Stroke complexity | 10 | 6 |
| Co-occurrence | Texture | 16 | 2 |
| 2×2 gram | Texture | 60 | 5 |
| Total | | 140 | 31 |

2 Handwriting Identification from Noisy Documents

The identification of handwriting from extremely noisy documents is a challenging task. Noise in document images often comes from two sources: 1) the physical degradation of the hardcopy document, and 2) the degradation introduced by digitalization. Both can affect the performance of document analysis systems significantly. A window-based morphological filtering is often used to remove noise [2, 3], and performs well for very small blocks, such as salt-pepper noise. In [1, 2] large blocks are treated as noise and removed based on heuristic rules. In our work we consider noise as a distinct class and use statistical pattern recognition techniques to identify it.

To handle the mis-classification, we exploit contextual information to refine the classification. Contextual information is very useful for improving the classification accuracy, since it uses the statistical dependency among neighboring components [4]. Among image models previously applied, the Markov Random Field (MRF) is well studied and has been successfully used in many applications. It can incorporate *a priori* contextual information in a quantitative way. In our work we use MRF to model the dependency among neighboring word blocks. Since noise has no concept of *word*, we use terminology *block* and *word* interchangeably.

2.1 Feature Extraction and Selection

We first segment the document at the word level using a connected component based approach [5], and then extract 140 features for each segmented word (Table 1). A large feature set may actually decrease the generality of the classifier when the number of the training samples is limited. The more features used, the more training samples needed, and the more expensive for feature extraction and classification. We use a straight forward feature selection technique to reduce the feature set. Our experiments show the best classification is achieved when only 31 features are selected, with an error rate of 5.7%. The last column of Table 1 lists the number of features selected in each category.

The detailed description of the feature extraction and selection can be found in [6].

2.2 Classification

We use Fisher classifier in our experiments, which projects the feature vector onto an axis so the ratio of between-class scatter to within-class scatter is maximized. Since the Fisher classifier is often used for two-class classification problems, we use three Fisher classifiers, each optimized for a two-class classification (printed text/handwriting, printed text/noise and handwriting/noise). Each classifier outputs a confidence of the classification and the final decision is made by fusing the outputs of all three

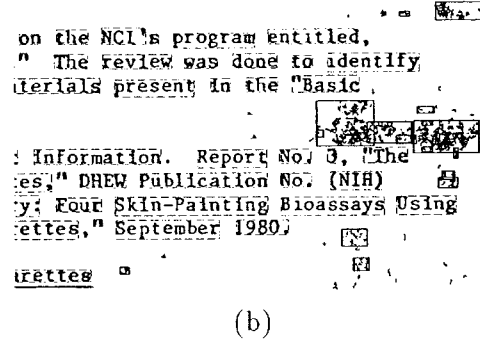
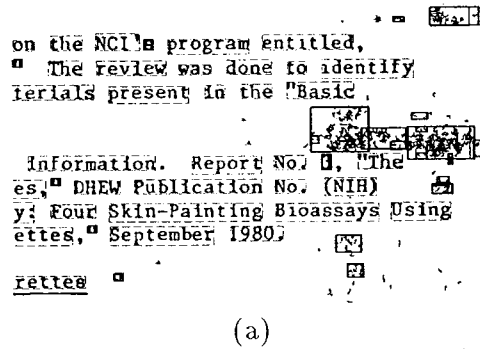


Figure 1. Word segmentation and classification. (a) Single word classification results; (b) post-processing results (mis-classifications of some punctuations in (a) have been corrected).

classifiers. If a classifier j is trained for class 1 and 2, then its output is not applicable to estimate the classification confidence of class 3. The final confidence is achieved by integrating the outputs from three classifiers as follows:

$$C_i = \frac{1}{3} \sum_{j=1}^3 C_{i,j} \quad (1)$$

$C_{i,j} \in [0, 1]$ for two applicable classifiers and $C_{i,j} = 0$ for the third classifier. So $C_i \in [0, 1]$.

Figure 1(a) shows the word segmentation and classification results for a document image, with solid, dot and dashed rectangles representing noise, handwriting, and printed text respectively. We can see some punctuations and small printed words are classified as noise. In next section we describe our approach to handle this problem.

2.3 MRF-Based Post-Processing

To refine the classification, we use a MRF-based scheme to rectify the mis-classification in Figure 1a. Let \underline{X} denote the random field defined on Ω and Γ denote the set of all possible configurations of \underline{X} on Ω . \underline{X} is the MRF with respect to the neighborhood η if:

$$\Pr(\underline{X} = \underline{x}) > 0 \quad \text{for all } \underline{x} \in \Gamma \quad (2)$$

$$P(\underline{x}_s / \underline{x}_r, r \in \Omega, r \neq s) = P(\underline{x}_s / \underline{x}_r, r \in \eta) \quad (3)$$

The establishment of the connection between MRF and Gibbs distribution provides ways for the optimization of the MRF. The problem of maximizing the *post-priori* probability of MRF converts to that of minimizing the total energy of the corresponding Gibbs distribution:

$$\hat{\underline{X}} = \arg \min_{\underline{X}} \sum_{c \in C} V_c(\underline{X}) \quad (4)$$

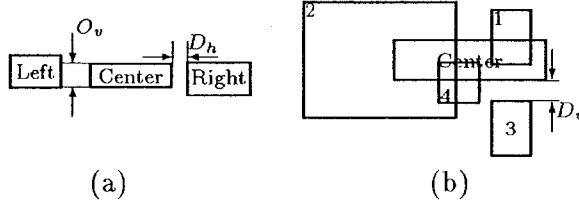


Figure 2. Clique definition. (a) C_p for horizontally arranged printed words; (b) C_n for noise blocks.

The value of clique potential $V_c(\underline{X})$ depends on the local configuration on the clique c .

We make use of the structure of documents and define two cliques for printed text and noise respectively. Printed words often form horizontal (or vertical) text lines. We define clique C_p as shown in Figure 2(a) for printed words, which models contextual constraints of printed words. Noise blocks exhibit rough random patterns and they tend to overlap. Figure 2(b) shows clique C_n defined primarily for noise blocks. From the definition of cliques we can refine the connection between two blocks, and then construct a graph in which nodes represent words and edges connect two connected blocks. The property of an edge can be measured by the distance $d(i, j)$ between two blocks. If a node is connected with more than one node on one side (left or right), we only keep the edge with the smallest distance.

We use the frequency of each clique in the training set to define the clique potentials, which can be expressed as a function of local conditional probabilities. Two clique potentials $V_p(c)$ and $V_n(c)$ for clique C_p and C_n are defined as:

$$V_p(c) = -\frac{P(X_l, X_c, X_r)}{(P(X_l)P(X_c)P(X_r))^w} \quad (5)$$

$$V_n(c) = -\frac{P(X_c, X_1, X_2, X_3, X_4)}{(P(X_c)P(X_1)P(X_2)P(X_3)P(X_4))^w} \quad (6)$$

Where X_l , X_c and X_r are labels for the left, center and right blocks of clique c ; w is a constant; and $X_i, i = 1, 2, 3, 4$ are labels for the corresponding i th nearest blocks. The energy of corresponding Gibbs distribution is:

$$U(\underline{X}/\underline{Y}) = -w_s \sum_{s \in \Omega} P(x_s/y_s) + w_p \sum_{c \in C_p} V_p(c) + w_n \sum_{c \in C_n} V_n(c) \quad (7)$$

where w_s , w_p and w_n are weights, which adjust the relative importance between classification confidence and contextual information of clique C_p and C_n . If w_s is fixed to 1, and w_p and w_n are set to 0, no contextual information is used. If we increase w_p and w_n , then more contextual information is emphasized. If we set w_p and w_n to infinity, or equivalently set $w_s = 0$, no classification confidence information is used.

2.4 Experiments

We collected 318 business letters from the tobacco industry litigation archives. These document images are noisy with a lot of handwritten annotations, a few logos, and no figures and tables. At current stage, we only identify three classes: printed text, handwriting and noise. Since the groundtruthing of each word block in the image would be extremely time consuming, we groundtruthed 94 of the noisy document images. These 94 images are used for testing, the remaining 224 images for training. There are about 1,500 handwritten words in the training set. Since there are many more printed text and noise elements, we randomly selected about 1,500 samples of each type to train three Fisher classifiers. We use *accuracy* and *precision* as metrics to evaluate the results:

Table 2. The classification results before and after post-processing.

| | Blocks | Before processing | | | After processing | | | Error Reduction Rate |
|--------------|--------|-----------------------------|----------|-----------|-----------------------------|----------|-----------|----------------------|
| | | Correctly classified blocks | Accuracy | Precision | Correctly classified blocks | Accuracy | Precision | |
| Printed text | 19,227 | 18,446 | 95.9% | 99.5% | 18,835 | 98.0% | 99.7% | 49.8% |
| Handwriting | 701 | 653 | 93.2% | 62.9% | 652 | 93.0% | 83.3% | -2.1% |
| Noise | 8,802 | 8,522 | 96.8% | 93.0% | 8,682 | 98.6% | 96.0% | 57.1% |
| Total | 28,730 | 2,7621 | 96.1% | N/A | 28,169 | 98.1% | N/A | 49.4% |

$$\text{Accuracy of type } i = \frac{\# \text{ of correctly classified blocks of type } i}{\# \text{ of blocks of type } i} \tag{8}$$

$$\text{Precision of type } i = \frac{\# \text{ of correctly classified blocks of type } i}{\# \text{ of blocks classified as type } i} \tag{9}$$

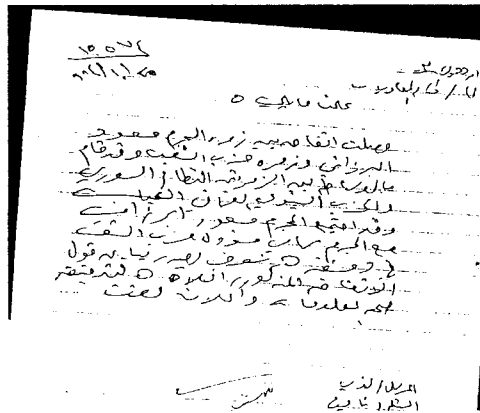
The results are listed in Table 2. The accuracy ranges from 93.2% to 96.8%, with the overall accuracy at 96.1%. While accuracy is very high, we notice precision for handwriting is very low. This is mainly because the number of handwritten words is very small, compared with printed text and noise. Therefore, even a small percentage of misclassification from printed text/noise to handwriting will significantly decrease its precision.

The results after we use the MRF are shown in the right columns of Table 2. The error rate of printed words and noise blocks is reduced to about half of the original. Although the error rate of handwriting increases slightly, the precision of handwriting increases from 62.9% to 83.3%, due to fewer mis-classifications of printed text and noise. The overall accuracy increases from 96.1% to 98.1%. Figure 1(b) shows the post-processing results of (b). We can see the mis-classified small printed words in Figure 1a are all corrected.

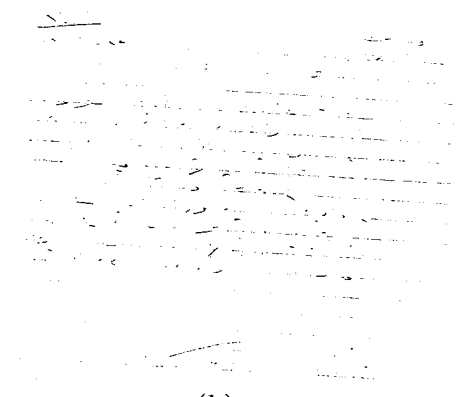
3 A Model-based Background Line Detection Algorithm

Line detection is widely used in table detection and interpretation [7, 8], engineering graph interpretation [9], and bank check/invoice processing [10]. The line detection algorithms presented can be broadly classified as Hough transform or vectorization based [11]. The Hough transform is a global approach with the ability to detect dashed and mildly broken lines, but is extremely time consuming [12]. To reduce the computation cost, a projection based method is proposed in [13] to search for lines only around 0° or 90°. The algorithm is much faster than Hough transform, however, it can only detect roughly horizontal or vertical lines. Vectorization based algorithms, such as BAG [7] and SPV methods [11], extract vectors from the image first, then merge vectors into lines. Recently Zheng presented a novel vectorization based algorithm called the Directional Single-Connected Chain (DSCC) method [8]. Each extracted DSCC represents a line segment and multiple non-overlapped DSCCs are merged into a line based on rules.

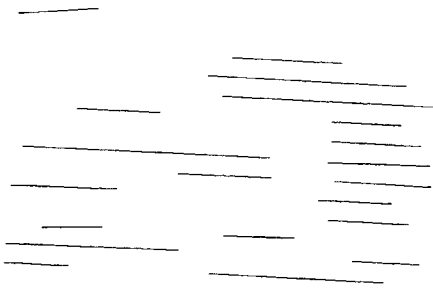
These line detection algorithms work well on relatively clean documents with solid or mildly broken lines. In our task there are two challenges: 1) the lines are severely broken due to the low image quality, and 2) the lines are mixed with text, making separation difficult. Figure 3(c) shows the line detection result using the DSCC algorithm. We can see only few lines are partially detected. It is very difficult, if not impossible, to detect these lines without contextual information. We present a novel model-based approach to systematically incorporate high level information to detect lines in the following sections.



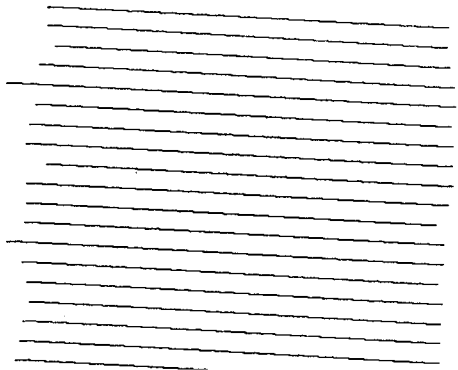
(a)



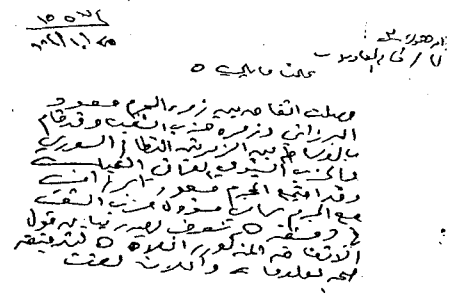
(b)



(c)



(d)



(e)

Figure 3. An example of background parallel line detection. (a) Original document image; (b) after filtering; (c) lines detected after merging neighboring DSCCs; (d) line detection results using the model; (e) after line removal.

3.1 Approach

We observed that 1) background lines are often parallel and 2) the vertical gaps between any two neighboring lines are roughly equal. We propose a stochastic model, $\mathcal{M}(\theta, y_1, y_2, \dots, y_N)$, for the line detection, where $y_i, i = 1, 2, \dots, N$ is the vertical position of the i^{th} background line, and N is the number of lines on the document. The model for a group of background lines is shown in Figure 4. We find a sequence can be modeled well by a HMM. The skew angle θ is estimated first. We then perform a coarse estimation of the vertical line gap \bar{g} , from the auto-correlation of the horizontal projection along the estimated skew angle. The Viterbi algorithm is used to search for the optimal position of background lines simultaneously in the projection profile. The estimation error of \bar{g} and variance between vertical line gaps are all compensated by the Viterbi decoding of the HMM model.

3.2 Pre-Processing

First we extract horizontal line segments using the DSCC based algorithm [8]. A horizontal DSCC is an array of connected vertical run-lengths, which can be a line segment, a text stroke or noise, for example. We only preserve those DSCCs with a small skew angle and a large aspect ratio, which are likely to be horizontal line segments. Figure 3(b) shows the image after DSCC filtering of the original image in Figure 3(a). We can see that most text strokes are filtered and the background line segments are well preserved. After filtering, we merge neighboring DSCCs into lines, as shown in Figure 3(c). Based on this initial detection result, we use a two-step coarse to fine method to estimate the skew angle θ . We then do a horizontal projection along the estimated angle. A coarse estimation of the average vertical line gap \bar{g} is calculated from the auto-correlation of the horizontal projection profile. The details are addressed in [14].

3.3 Model-based Line Detection

3.3.1 HMM Model for Line Position

A HMM model includes five elements:

1. N , the number of the states in the model.
2. M , the number of distinct observation symbols per state.
3. The state transition probability distribution matrix:
 $A = \{a_{ij}\}$.
4. The probability distribution matrix of the observation symbol:
 $B = \{b_{ij}\}$.
5. The initial state distribution π .

HMM models have been successfully used in speech recognition [15] and handwriting recognition [16]. In our approach we use it to model the sequence of vertical line position y_i .

$$P(Y_i|Y_1, Y_2, \dots, Y_{i-1}) = P(Y_i|Y_{i-1}) \quad (10)$$

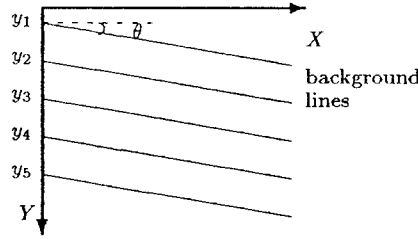


Figure 4. A group of background lines.

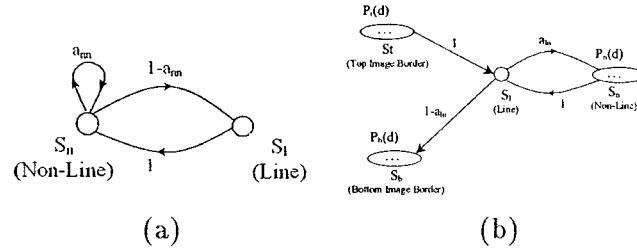


Figure 5. HMM Models for the horizontal projection profile. (a) A standard HMM model; (b) A HMM model with explicit duration.

We use uppercase characters to represent random variables (Y_i), and lowercase characters to represent the value of the random variables (y_i). The actual y_i is not observable. Instead we can only get the horizontal projection profile $h_k, k = 1, 2, \dots, T$, where T is the dimension of the profile.

$$\begin{aligned}
 &P(H_i | Y_1 = y_1, \dots, Y_N = y_N) \\
 &= \begin{cases} P(H_k | \exists i, k = y_i) & \text{A line is on } k \\ P(H_k | \forall i, k \neq y_i) & \text{No lines are on } k \end{cases} \quad (11)
 \end{aligned}$$

We define two states: line state S_l and non-line state S_n . A standard HMM model for our problem is shown in Figure 5(a). To model the top and bottom image borders, we add two states: S_t for top image border and S_b for bottom image border. The model with explicit state duration is shown in Figure 5(b).

The state transition probability is simple and labeled on Figure 5(b). We set a_{ln} to 0.5, and the detection result is not sensitive to a_{ln} . The observation is the projection profile h_k , which takes values between $[0, w]$, where w is the width of the image. We quantize h_k to five levels: $0, w/16, w/8, w/4, w$. The observation probability distribution matrix B is estimated from the groundtruthed background lines.

3.3.2 Decoding of a HMM model

Given the observation sequence $O = h_k, k = 1, 2, \dots, T$, and the HMM model λ , we want to search for an optimal state sequence $Q = q_1 q_2 \dots q_T$, to maximize $P(Q|O, \lambda)$, which is equivalent to maximizing $P(Q, O|\lambda)$. We use the Viterbi algorithm to decode the HMM models. For each $t, t = 1, \dots, T$, the algorithm remembers the best decoding path at time t . Therefore, the best decoding sequence q_1, q_2, \dots, q_T is achieved after the decoding process is finished.

The HMM parameters estimated directly from the groundtruthed database is not optimal since: 1) the data is sparse, and some entries do not appear, or only appear very few times in the training set; and 2) the Viterbi algorithm searches the hidden state sequence with the highest probability, given the observation sequence and the model. The optimization criteria does not minimize the final detection error, especially when the model mis-matches.

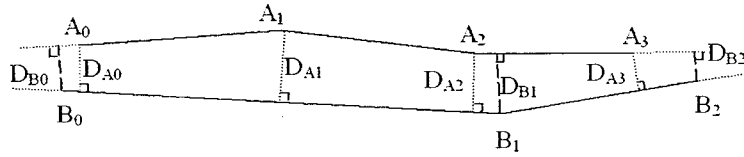


Figure 6. Vertical distance between two polylines.

To reduce the effect of sparse data, we smooth the duration distribution $p_n(d)$ of state S_n . Suppose the state duration is symmetric around the average vertical line gap, we perform the following averaging:

$$p_n(\bar{g} + i) = p_n(\bar{g} - i) = \frac{p_n(\bar{g} + i) + p_n(\bar{g} - i)}{2} \quad (12)$$

After averaging, we set the empty entries to the minimal value of all non-zero entries. We then use simplex search method proposed by Nelder and Mead to minimize the detection error [17].

3.3.3 Post-Processing

An ideal straight line can be presented with two parameters a and b as:

$$y = a \times x + b \quad (13)$$

However, due to the distortion introduced by the photocopying and scanning, some background lines curve, and can not be described well with two end points. The maximal presentation error may be up to 10 pixels, much larger than the typical line width of 2-4 pixels. Therefore, a polyline representation is exploited to present a real line in our following evaluation experiments, as shown in Figure 6. After splitting a line into several segments, we can present a line with a sequence of points (P_0, P_1, \dots, P_m) . Experiments show 2 or 3 segments are sufficient to represent most distorted background lines.

3.4 Experiments

3.4.1 Evaluation Protocol

Line detection accuracy can be evaluated at the pixel level and the line level [18]. The pixel level evaluation compares the difference of the pixels between groundtruthed and detected lines. It is straightforward and objective, but groundtruthing at the pixel level is extremely expensive when lines are broken, distorted and overlapped with text. Therefore, we evaluate the algorithm at the line level. Two metrics, the vertical and horizontal distance, are defined. Vertical distance is defined as a modified Hausdorff distance. The Hausdorff distance between two point sets is:

$$H(A, B) = \max\{h(A, B), h(B, A)\} \quad (14)$$

where,

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\| \quad (15)$$

and $\|\cdot\|$ is some underlying norm (e.g., the L_2 or Euclidean distance).

The horizontal distance, $hd(A, B)$, and horizontal matching rate are defined to measure the detection accuracy of left and right end points. Suppose the horizontal coordinates of the left and right end points

Table 3. Background line detection result.

| | Ground-truthed Lines | Detected Lines | Correct | Partial Correct | Missed | False Alarm |
|--------------|----------------------|----------------|------------------|-----------------|-------------|--------------|
| Training Set | 2,274 | 2,319 | 2,212 (97.3%) | 56 (2.5%) | 6 (0.3%) | 51 (2.2%) |
| Testing Set | 1,596 | 1,631 | 1,545 (96.8%) | 49 (3.0%) | 2 (0.1%) | 37 (2.3%) |

of polyline A and B are (L_A, R_A) and (L_B, R_B) respectively. Then, the horizontal distance $hd(A, B)$ are defined as:

$$hd(A, B) = \frac{|L_A - L_B| + |R_A - R_B|}{2} \quad (16)$$

And the horizontal matching rate is defined as:

$$m = \frac{\min\{R_A, R_B\} - \max\{L_A, L_B\}}{\max\{R_A, R_B\} - \min\{L_A, L_B\}} \quad (17)$$

A detected line matches a groundtruthed line if the vertical distance is less than $\bar{g}/3$. If there is more than one detected line matching a groundtruthed line, or vice versa, then only the match with the minimal distance is kept. The vertical detection distance of a groundtruthed line is defined as the vertical distance to its matched detected line. If a groundtruthed line can not match a detected line, it is mis-detected. For a matched groundtruthed line, if the detection distance is within 5 pixels, then it is said to be detected correctly. Otherwise it is regarded as partially detected. A false alarm happens if a detected line can not match any groundtruthed line. The false alarm rate is calculated as the number of false alarm lines over the number of groundtruth lines.

3.4.2 Experimental Results

We obtained 168 Arabic document images with a total of 3,870 groundtruthed lines, most of which are severely broken. 100 images are used to train the parameters of the HMM model; and the remaining 68 images are taken as the testing set. The detection results are shown in Table 3. On the testing set, 96.8% lines are detected correctly and only 2 lines are missed. The false alarm rate is about 2.3%. Most of the false alarms are generated because our model detected severely broken lines which are not groundtruthed based on the subjective judgment of the groundtruther. Our ultimate goal is to remove background lines to achieve a clean document. These false alarms do not create problems for line removal, as very few pixels are removed. Therefore, we adjust the parameter to keep mis-detection rate low.

Figure 3(d) shows the model based line detection result. Compared with Figure 3(c), we can see with context information we can get much better result. In the example of Figure 7(a), we remove 35 rows of the image (about half of the average vertical line gap of this document). The corresponding line detection result is shown in Figure 7(b). Our algorithm only missed one line, since the variance of the vertical line gap is outside of the model parameter, which is $[-11, 11]$ pixels. Our HMM can cope with up to 11 pixels vertical line gap variance, and the post-processing module can deal with variance of some more pixels. Therefore, all other lines are detected correctly.

Our final experiment was to test the robustness of our algorithm. We selected a document with background lines of relatively good quality without too much pixel loss. We manually separate the document into two layers: text and background lines, as shown in Figure 8(b) and (c) respectively. We randomly flip some black pixels of background lines to white, and then merge the degraded background lines with the original text. Figure 8(d) and (f) are the degraded images with only about 10% and 5%

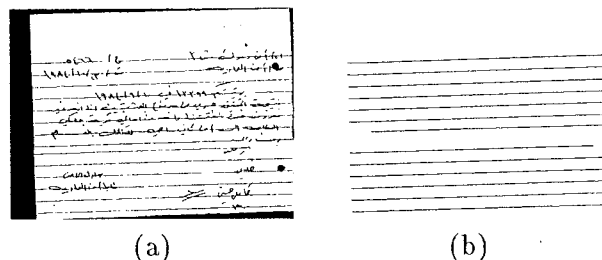


Figure 7. An example of a model mis-match. (a) A document image with 35 image rows removed; (b) Line detection result of (a).

pixels in the lines preserved. With 10% pixels preserved, our algorithm can still detect all background lines, as shown in Figure 8(e). When only about 5% pixels preserved, our algorithm breaks down, as shown in Figure 8(h). The most vulnerable parts are the skew angle and the average vertical line gap estimation. When 5% line pixels preserved, the vertical line gap is estimated twice as large as the actual value. Therefore, about half of the background lines are missed.

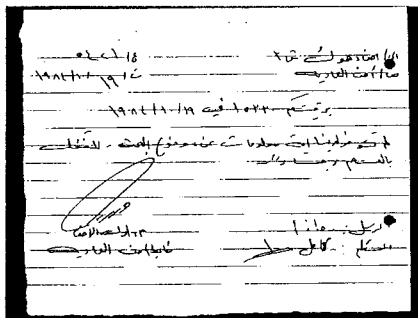
After line detection, we can remove these detected lines to achieve a cleaned version of the document. Figure 3(e) is the result of Figure 3(a) after we remove the black pixels on the line and filter the noise. While the result is encouraging, we find some text strokes touching the detected lines are removed erroneously. We are investigating a more robust algorithm to remove the lines and reserve the text strokes.

4 Conclusion and Future Work

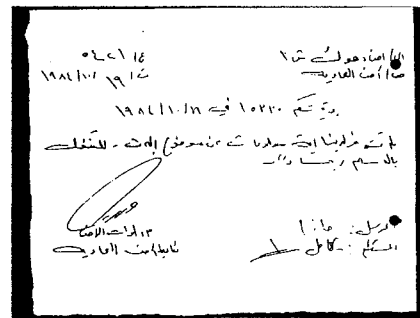
In this paper, we have presented an overview of our initial work on processing noisy documents. To identify handwriting from extremely noisy document images, we take noise as a distinct class, and use statistical classification techniques to classify each word block into: printed text, handwriting and noise. After single word classification, a MRF is used to incorporate contextual information to refine the classification results. Experiments show the MRF is very effective to model local dependency among neighboring image components. To detect severely broken lines, we proposed a model based approach to incorporate high level constraints into a general line detection algorithm. Experiments show our method can detect 94% lines in the database we collected. Our next focus includes removing the detected background lines touching the text, and indexing signatures and logos efficiently in large document databases.

References

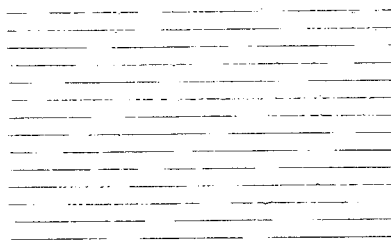
- [1] A. K. Jain and B. Yu. Document representation and its application to page decomposition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(3):294–308, 1998.
- [2] L. O’Gorman. Image and document processing techniques for the RightPages electronic library system. In *Proc. Int’l Conf. Pattern Recognition*, pages 820–825, 1992.
- [3] R. P. Loce and E. R. Dougherty. *Enhancement and Restoration of Digital Documents – Statistical Design of Nonlinear Algorithms*. SPIE Optical Engineering Press, 1997.
- [4] J. J. Hull. Incorporating language syntax in visual text recognition with a statistical model. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(12):1251–1256, 1996.
- [5] Y. Zheng, H. Li, and D. Doermann. The segmentation and identification of handwriting in noisy document images. In *Proc. Document Analysis System*, pages 95–105, 2002.
- [6] Y. Zheng, H. Li, and D. Doermann. Machine printed text and handwriting identification in noisy document images. Technical report, LAMP Lab, University of Maryland, Collge Park, 2002.



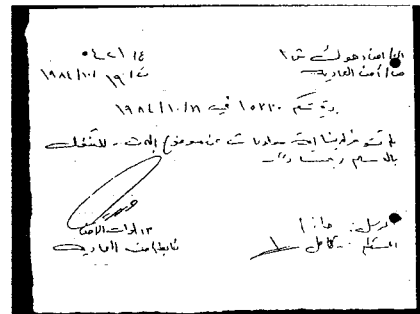
(a)



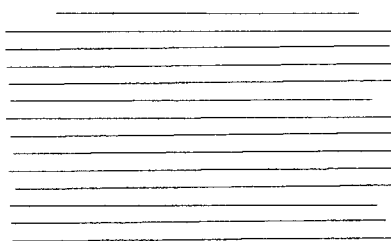
(b)



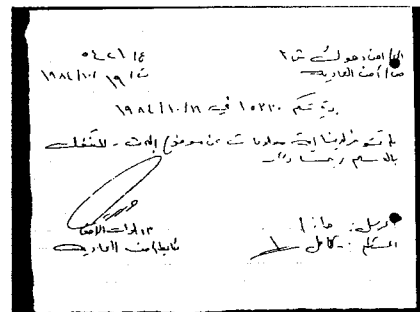
(c)



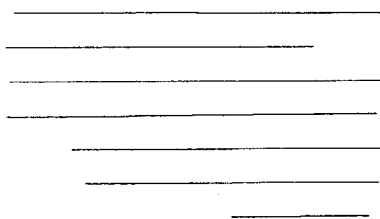
(d)



(e)



(f)



(h)

Figure 8. An experiment testing how document degradation affects line detection algorithm. (a) Original image, which is separated into two layers: text (b) and background lines (c) manually; (d) About 10% pixels of the background lines are preserved; (e) Line detection result of (d); (f) About 5% pixels of the background lines are preserved; (h) Line detection result of (f).

- [7] B. Yu and A. K. Jain. A generic system for form dropout. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(11):1127–1131, 1996.
- [8] Y. Zheng, C. Liu, and X. Ding. Form frame line detection with directional single-connected chain. In *Proc. Int'l Conf. Document Analysis and Recognition*, pages 699–703, 2001.
- [9] D. Dori, Y. Liang, and J. Dowell. Sparse-pixel recognition of primitives in engineering drawings. *Machine Vision and Application*, 6:69–82, 1993.
- [10] Y. Y. Tang, C. Y. Suen, and C. D. Yan. Financial document processing based on staff line and description language. *IEEE Trans. Systems, Man and Cybernetics*, 25(5):738–753, 1995.
- [11] W. Liu and D. Dori. From raster to vectors: Extracting visual information from line drawings. *Pattern Analysis and Application*, 2(1):10–21, 1999.
- [12] J. Illingworth and J. Kittler. A survey of the Hough transform. *CVGIP*, 44:87–116, 1988.
- [13] J. Liu, X. Ding, and Y. Wu. Description and recognition of form and automated form data entry. In *Proc. Int'l Conf. Document Analysis and Recognition*, pages 579–582, 1995.
- [14] Y. Zheng, H. Li, and D. Doermann. A model-based line detection algorithm in documents. In *Proc. Int'l Conf. Document Analysis and Recognition*, (submitted), 2003.
- [15] L. R. Rabiner. A tutorial on hidden markov models and selected application in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [16] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(1):63–84, 2000.
- [17] J. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.
- [18] W. Liu and D. Dori. A protocol for performance evaluation of line detection algorithms. *Machine Vision and Application*, 9(5/6):57–68, 1997.

Summarizing Noisy Documents

Hongyan Jing

IBM T.J. Watson Research Center
Yorktown Heights, NY
hjing@us.ibm.com

Daniel Lopresti

19 Elm Street
Hopewell, NJ
dpl@dlopresti.com

Chilin Shih

150 McMane Avenue
Berkeley Heights, NJ
cls@prosodies.org

Abstract

We investigate the problem of summarizing text documents that contain errors as a result of optical character recognition. Each stage in the process is tested, the error effects analyzed, and possible solutions suggested. Our experimental results show that current approaches, which are developed to deal with clean text, suffer significant degradation even with slight increases in the noise level of a document. We conclude by proposing possible ways of improving the performance of noisy document summarization.

1 Introduction

Summarization aims to provide a user with the most important information gleaned from a document (or collection of related documents) [16]. A good summary can help the reader grasp key subject matter without requiring study of the entire document. This is especially useful nowadays as information-overload becomes a serious issue.

Much attention is currently being directed towards the problem of summarization [6, 25]. However, the focus to date has typically been on clean, well-formatted documents, i.e., documents that contain relatively few spelling and grammatical errors, such as news articles or published technical material. In this paper, we present a pilot study of noisy document summarization, motivated primarily by the impact of various kinds of physical degradation that pages may endure before they are scanned and processed using optical character recognition (OCR) software.

Understandably, summarizing documents that contain many errors is an extremely difficult task. In our study, we focus on analyzing how the quality of summaries is affected by the level of noise in the input document, and how each stage in summarization is impacted by the noise. Based on our analysis, we suggest possible ways of improving the performance of automatic summarization systems for noisy documents. We hope to use what we have learned from

this initial investigation to shed light on the directions future work should take.

What we ascertain from studying the problem of noisy document summarization can be useful in a number of other applications as well. Noisy documents constitute a significant percentage of documents we encounter in everyday life. The output from OCR and speech recognition (ASR) systems typically contain various degrees of errors, and even purely electronic media, such as email, are not error-free. To summarize such documents, we need to develop techniques to deal with noise, in addition to working on the core algorithms. Whether we can successfully handle noise will greatly influence the final quality of summaries of such documents.

A number of researchers have begun studying problems relating to information extraction from noisy sources. To date, this work has focused predominately on errors that arise during speech recognition, and on problems somewhat different from summarization. For example, Gotoh and Renals propose a finite state modeling approach to extract sentence boundary information from text and audio sources, using both n-gram and pause duration information [8]. They found that precision and recall of over 70% could be achieved by combining both kinds of features. Palmer and Ostendorf describe an approach for improving named entity extraction by explicitly modeling speech recognition errors through the use of statistics annotated with confidence scores [20]. Hori and Furui summarize broadcast news speech by extracting words from automatic transcripts using a word significance measure, a confidence score, linguistic likelihood, and a word concatenation probability [11].

There has been much less work, however, in the case of noise induced by optical character recognition. Early papers by Taghva, et al. show that moderate error rates have little impact on the effectiveness of traditional information retrieval measures [23, 24], but this conclusion does not seem to

apply to the task of summarization. Miller, et al. study the performance of named entity extraction under a variety of scenarios involving both ASR and OCR output [18], although speech is their primary interest. They found that by training their system on both clean and noisy input material, performance degraded linearly as a function of word error rates. They also note in their paper: "To our knowledge, no other information extraction technology has been applied to OCR material" (pg. 322).

An intriguing alternative to text-based summarization is Chen and Bloomberg's approach to creating summaries without the need for optical character recognition [4]. Instead, they extract indicative summary sentences using purely image-based techniques and common document layout conventions. While this is effective when the final summary is to be viewed on-screen by the user, the issue of optical character recognition must ultimately be faced in most applications of interest (e.g., keyword-driven information retrieval).

For the work we present in this paper, we performed a small pilot study in which we selected a set of documents and created noisy versions of them. These were generated both by OCR'ing real pages and by using a filter we have developed that injects various levels of noise into an original source document. The clean and noisy documents were then piped through a summarization system. We tested different modules that are often included in such systems, including sentence boundary detection, part-of-speech tagging, syntactic parsing, extraction, and editing of extracted sentences. The experimental results show that these modules suffer significant degradation as the noise level in the document increases. We discuss the errors made at each stage and how they affect the quality of final summaries.

In Section 2, we describe our experiment, including the data creation process and various tests we performed. In Section 3, we analyze the results of the experiment and correlate the quality of summaries with noise levels in the input document and the errors made at different stages of the summarization process. We then discuss some of the challenges in summarizing noisy documents and suggest possible methods for improving the performance of noisy document summarization. We conclude with a proposal for future work.

2 The Experiment

2.1 Data Creation

We selected a small set of four documents to study in our experiment. Three of four documents were from the data collection used in the Text REtrieval Conferences (TREC) [10] and one was from a Telecom-

munications corpus we collected ourselves [13]. All were professionally written news articles, each containing from 200 to 800 words (the shortest document was 9 sentences and the longest was 38 sentences).

For each document, we created 10 noisy versions. The first five corresponded to real pages that had been printed, possibly subjected to a degradation, scanned at 300 dpi using a UMAX Astra 1200S scanner, and then OCR'ed with Caere OmniPage Limited Edition. These included:

clean The page as printed.

fax A faxed version of the page.

dark An excessively dark (but legible) photocopy.

light An excessively light (but legible) photocopy.

skew The clean page skewed on the scanner glass.

Note that because the faxed and photocopied documents were processed by running them through automatic page feeders, these pages can also exhibit noticeable skew. The remaining five sample documents in each case were electronic copies of the original that had had synthetic noise (single-character deletions, insertions, and substitutions) randomly injected at predetermined rates: 5%, 10%, 15%, 20%, and 25%.

A summary was created for each document by human experts. For the three documents from the TREC corpus, the summaries were generated by taking a majority opinion. Each document was given to five people who were asked to select 20% of the original sentences as the summary. Sentences selected by three or more of the five human subjects were included in the summary of the document. (These summaries were created for our prior experiments studying summarization evaluation methodologies [14].) For the document from the Telecommunications corpus, an abstract of the document was provided by a staff writer from the news service. These human-created summaries were useful in evaluating the quality of the automatic summaries.

2.2 Summarization Pipeline Stages

We are interested in testing how each stage of a summarization system is affected by noise, and how this in turn affects the quality of the summaries. Many summarization approaches exist, and it would be difficult to study the effects of noise on all of them. However, the following pipeline is common to many summarization systems:

- Step 1: Tokenization. The main task here is to break the text into sentences. Tokens in the input text are also identified.

- Step 2: Preprocessing. This typically involves part-of-speech tagging and syntactic parsing. This step is optional; some systems do not perform tagging and parsing at all. Topic segmentation is deployed by some summarization systems, but not many.
- Step 3: Extraction. This is the main step in summarization, in which the automatic summarizer selects key sentences (sometimes paragraphs or phrases) to include in the summary. Many different approaches for sentence extraction have been proposed and various types of information are used to find summary sentences, including but not limited to: frequency, lexical cohesion, sentence position, cue phrases, discourse structures, and overlapping information in multiple documents.
- Step 4: Editing. Some systems post-edit the extracted sentences to make them more coherent and concise.

For each stage in the pipeline, we selected one or two systems that perform the task and tested their performance on both clean and noisy documents.

- For tokenization, we tested two tokenizers: one is a rule-based system that decides sentence boundaries based on heuristic rules encoded in the program, and the other one is a trainable tokenizer that uses a decision tree approach for detecting sentence boundaries and has been trained on a large amount of data.
- For part-of-speech tagging and syntactic parsing, we tested the English Slot Grammar (ESG) parser [17]. The outputs from both tokenizers were tested on ESG.
- For extraction, we used a program that relies on lexical cohesion, frequency, sentence positions, and cue phrases to identify key sentences [13]. The length parameter of the summaries was set to 20% of the number of sentences in the original document. The output from the rule-based tokenizer was used in this step.
- In the last step, we tested a cut-and-paste system that edits extracted sentences by simulating the revision operations often performed by professional abstractors [13]. The outputs from the three previous steps were used by the cut-and-paste system.

All of the summaries produced in this experiment were generic, single-document summaries (i.e., the summary was about the main topic conveyed

in a document, rather than some specific information that is relevant to particular interests defined by a user). Multiple document summaries are more complex, and we did not study them in this experiment. Neither did we study translingual or query-based summarization. However, we are very interested in studying translingual, multi-document, or query-based summarization of noisy documents in the future.

3 Results and Analysis

In this section, we present results at each stage of summarization, analyzing the errors made and their effects on the quality of summaries.

3.1 OCR performance

We begin by examining the overall performance of the OCR process. Using standard edit distance techniques [7], we can compare the output of OCR to the ground-truth to classify and quantify the errors that have arisen. We then compute, on a per-character and per-word basis, a figure for average precision (percentage of characters or words recognized that are correct) and recall (percentage of characters or words in the input document that are correctly recognized). As indicated in Table 1, OCR performance varies widely depending on the type of degradation. Precision values are generally higher than recall because, in certain cases, the OCR system failed to produce output for a portion of the page in question. Since we are particularly interested in punctuation due to its importance in delimiting sentence boundaries, we tabulate a separate set of precision and recall values for such characters. Note that these are uniformly lower than the other values in the table. Recall, in particular, is a serious issue; many punctuation marks are missed in the OCR output.

3.2 Sentence boundary errors

Since most summarization systems rely on sentence extraction, it is important to identify sentence boundaries correctly. For clean text, sentence boundary detection is not a big problem; the reported accuracy is usually above 95% [19, 21, 22]. However, since such systems typically depend on punctuation, capitalization, and words immediately preceding and following punctuation to make judgments about potential sentence boundaries, detecting sentence boundaries in noisy documents is a challenge due to the unreliability of such features. Punctuation errors arise frequently in the OCR'ing of degraded page images, as we have just noted.

We tested two tokenizers: one is a rule-based system that relies on heuristics encoded in the program, and the other is a decision tree system that has been

Table 1: OCR performance relative to ground-truth (average precision and recall).

| | Per-Character | | | | Per-Word | |
|-----------|---------------|--------|-------------|--------|----------|--------|
| | All Symbols | | Punctuation | | Prec. | Recall |
| | Prec. | Recall | Prec. | Recall | | |
| OCR.clean | 0.990 | 0.882 | 0.869 | 0.506 | 0.963 | 0.874 |
| OCR.light | 0.897 | 0.829 | 0.556 | 0.668 | 0.731 | 0.679 |
| OCR.dark | 0.934 | 0.739 | 0.607 | 0.539 | 0.776 | 0.608 |
| OCR.fax | 0.969 | 0.939 | 0.781 | 0.561 | 0.888 | 0.879 |
| OCR.skew | 0.991 | 0.879 | 0.961 | 0.496 | 0.963 | 0.869 |

trained on a large amount of data. We are interested in how well these systems perform on noisy documents and the kinds of errors they make.

The experimental results show that for the clean text, the two systems perform almost equally well. We manually checked the results for the four documents and found that both tokenizers made very few errors. There should be 90 sentence boundaries in total. The decision tree tokenizer correctly identified 88 of the sentence boundaries and missed two. The rule-based tokenizer correctly identified 89 of the boundaries and missed one. Neither system made any false positive errors (i.e., they did not break sentences at non-sentence boundaries).

For the noisy documents, however, both tokenizers made significant numbers of errors. The types of errors they made, moreover, were quite different. While the rule-based system made many false negative errors, the decision tree system made many false positive errors. Therefore, the rule-based system identified far fewer sentence boundaries than the truth, while the decision tree system identified far more than the truth.

Table 2 shows the number of sentences identified by each tokenizer for different versions of the documents. As we can see from the table, the noisier the documents, the more errors the tokenizers made. This relationship was demonstrated clearly by the results for the documents with synthetic noise. As the noise rate increases, the number of boundaries identified by the decision tree tokenizer gradually increases, and the number of boundaries identified by the rule-based tokenizer gradually decreases. Both numbers diverge from truth, but they err in opposite directions.

The two tokenizers behaved less consistently on the OCR'ed documents. For OCR.light, OCR.dark, and OCR.fax, the decision tree tokenizer produced more sentence boundaries than the rule-based tokenizer. But for OCR.clean and OCR.skew, the decision tree tokenizer produced fewer sentence boundaries. This may be related to the noise level in the document. OCR.clean and OCR.skew contain

fewer errors than the other noisy versions (recall Table 1). According to our computations, 97% of the words that occurred in OCR.clean or OCR.skew also appeared in the original document, while other OCR'ed documents have a much lower word overlap, as shown in Table 4. This seems to indicate that the decision tree tokenizer tends to identify fewer sentence boundaries than the rule-based tokenizer for clean text or documents with very low levels of noise, but more sentence boundaries when the documents have a relatively high level of noise.

Errors made at this stage are extremely detrimental, since they will propagate to all of the other modules in a summarization system. When a sentence boundary is incorrectly marked, the part-of-speech tagging and the syntactic parsing are likely to fail. Sentence extraction may become problematic; for example, one of the documents in our test set contains 24 sentences, but for one of its noisy versions (OCR.dark), the rule-based tokenizer missed most sentence boundaries and divided the document into only three sentences, making extraction at the sentence level difficult at best.

Since sentence boundary detection is important to summarization, the development of robust techniques that can handle noisy documents is worthwhile. We will return to this point in Section 4.

3.3 Parsing errors

Some summarization systems use a part-of-speech tagger or a syntactic parser in their preprocessing steps. To study the errors made at this stage, we piped the results from both tokenizers to the ESG parser, which requires as input divided sentences and returns a parse tree for each input sentence. The parse tree also includes a part-of-speech tag for each word in the sentence.

We computed the percentage of sentences that ESG failed to return a complete parse tree, and used that value as one way of measuring the performance of the parser on the noisy documents. As we can see from Table 3, a significant percentage of noisy sentences were not parsed. Even for the documents

Table 2: Sentence boundary detection results: total number of sentences detected and average words per sentence for two tokenizers. The ground-truth is represented by *Original*.

| | Tokenizer 1 (Decision tree) | | Tokenizer 2 (Rule-based) | |
|-----------|-----------------------------|------------------|--------------------------|------------------|
| | Sentences | Avg. words/sent. | Sentences | Avg. words/sent. |
| Original | 88 | 23 | 89 | 22 |
| Snoise.05 | 95 | 20 | 70 | 27 |
| Snoise.10 | 97 | 20 | 69 | 28 |
| Snoise.15 | 105 | 19 | 65 | 30 |
| Snoise.20 | 109 | 17 | 60 | 31 |
| Snoise.25 | 121 | 15 | 51 | 35 |
| OCR.clean | 77 | 23 | 82 | 21 |
| OCR.light | 119 | 15 | 64 | 28 |
| OCR.dark | 70 | 21 | 46 | 33 |
| OCR.fax | 78 | 26 | 75 | 27 |
| OCR.skew | 77 | 23 | 82 | 21 |

with synthetic noise at a 5% rate, around 60% of the sentences cannot be handled by the parser. For the sentences that were handled, the returned parse trees may not be correct. For example, the sentence “*Internet sites found that almost 90 percent collected personal information from youngsters*” was transformed to “*uInternet sites fo6ndha alQmostK0 percent coll / 9ed pe?*” after adding synthetic noise at a 25% rate. For this noisy sentence, the parser returned a complete parse tree that marked the word “*sites*” as the main verb of the sentence, and tagged all the other words in the sentence as nouns.¹ Although a complete parse tree is returned in this case, it is incorrect. This may explain the phenomenon that the parser returned a higher percentage of complete parse trees for documents with synthetic noise at the 25% rate than for documents with lower levels of noise.

The above results indicate that syntactic parsers may be very vulnerable to noise in a document. Even low levels of noise tend to lead to a significant drop in performance. For documents with high levels of noise, it may be better not to rely on syntactic parsing at all since it will likely fail on a large portion of the text, and even when results are returned, they will be unreliable.

3.4 Extract quality versus noise level

In the next step, we studied how the sentence extraction module in a summarization system is affected by noise in the input document. For this, we used a sentence extraction system we had developed previously [13]. The sentence extractor relies on lexical links between words, word frequency, cue phrases, and sentence positions to identify key sentences. We

¹One reason might be that the tagger is likely to tag unknown words as nouns, since most out-of-vocabulary words are nouns.

Table 3: Percentage of sentences with incomplete parse trees from the ESG parser. Sentence boundaries were first detected using Tokenizer 1 and Tokenizer 2, and divided sentences were given to ESG as input.

| | Tokenizer 1 | Tokenizer 2 |
|-----------|-------------|-------------|
| Original | 10% | 5% |
| Snoise.05 | 59% | 58% |
| Snoise.10 | 69% | 71% |
| Snoise.15 | 66% | 81% |
| Snoise.20 | 64% | 66% |
| Snoise.25 | 58% | 76% |
| OCR.clean | 2% | 3% |
| OCR.light | 46% | 53% |
| OCR.dark | 37% | 43% |
| OCR.fax | 37% | 30% |
| OCR.skew | 5% | 6% |

set the summary length parameter as 20% of the number of sentences in the original document. This sentence extraction system does not use results from part-of-speech tagging or syntactic parsing, only the output from the rule-based tokenizer.

Evaluation of noisy document summaries is an interesting problem. Intrinsic evaluation (i.e., asking human subjects to judge the quality of summaries) can be used, but this appears much more complex than intrinsic evaluation for clean documents. When the noise rate in a document is high, even when a summarization system extracts the right sentences, a human subject may still rank the quality of the summary as very low due to the noise. Extrinsic evaluation (i.e., using the summaries to perform certain tasks and measuring how much the summaries help in performing the tasks) is also difficult since the

noise level of extracted sentences can significantly affect the result.

We employed three measures that have been used in the Document Understanding Conference [6] for assessing the quality of generated summaries: unigram overlap between the automatic summary and the human-created summary, bigram overlap, and the simple cosine. These results are shown in Table 4. The unigram overlap is computed as the number of unique words occurring both in the extract and the ideal summary for the document, divided by the total number of unique words in the extract. Bigram overlap is computed similarly, replacing words with bigrams. The simple cosine is computed as the cosine of two document vectors, the weight of each element in the vector being $1/\sqrt{N}$, where N is the total number of elements in the vector.

Not surprisingly, summaries of noisier documents generally have a lower overlap with human-created summaries. However, this can be caused by either the noise in the document or poor performance of the sentence extraction system. To separate these effects and measure the performance of sentence extraction alone, we also computed the unigram overlap, bigram overlap, and cosine between each noisy document and its corresponding original text. These numbers are included in Table 4 in parentheses; they are an indication of the average noise level in a document. For instance, the table shows that 97% of words that occurred in OCR.clean documents also appeared in the original text, while only 62% of words that occurred in OCR.light appeared in the original. This confirms that OCR.clean is less noisy than OCR.light.

3.5 Abstract generation for noisy documents

To generate more concise and coherent summaries, a summarization system may edit extracted sentences. To study how this step in summarization is affected by noise, we tested a cut-and-paste system that edits extracted sentences by simulating revision operations often used by human abstractors, including the operations of removing phrases from an extracted sentence, and combining a reduced sentence with other sentences. This cut-and-paste stage relies on the results from sentence extraction in the previous step, the output from ESG, and a co-reference resolution algorithm.

For the clean text, the cut-and-paste system performed sentence reduction on 59% of the sentences that were extracted in the sentence extraction step, and sentence combination on 17% of the extracted sentences. For the noisy text, however, the system applied very few revision operations to the extracted (noisy) sentences. Since the cut-and-paste system

relies on the output from ESG and co-reference resolution, which failed on most of the noisy text, it is not surprising that it did not perform well under these circumstances. Editing sentences requires a deeper understanding of the document and, as the last step in the summarization pipeline, relies on results from all of the previous steps. Hence, it is affected most severely by noise in the input document.

4 Challenges in Noisy Document Summarization

In the previous section, we have presented and analyzed errors at each stage of summarization when applied to noisy documents. The results show that the methods we tested at every step are fragile, susceptible to failures and errors even with slight increases in the noise level of a document. Clearly, much work needs to be done to achieve acceptable performance in noisy document summarization. We need to develop summarization algorithms that do not suffer significant degradation when used on noisy documents. We also need to develop the robust natural language processing techniques that are required by summarization. For example, sentence boundary detection systems that can reliably identify sentence breaks in noisy documents are clearly important. One way to achieve this might be to retrain an existing system on noisy documents so that it will be more tolerant of noise. However, this is only applicable if the noise level is low. Significant work is needed to develop robust methods that can handle documents with high noise levels.

In the remainder of this section, we discuss several issues in noisy document summarization, identifying the problems and proposing possible solutions. We regard this as a first step towards a more comprehensive study on the topic of noisy document summarization.

4.1 Choosing an appropriate granularity

It is important to choose an appropriate unit level to represent the summaries. For clean text, sentence extraction is a feasible goal since we can reliably identify sentence boundaries. For documents with very low levels of noise, sentence extraction is still possible since we can probably improve our programs to handle such documents. However, for documents with relatively high noise rates, we believe it is better to forgo sentence extraction and instead favor extraction of keywords or noun phrases, or generation of headline-style summaries. In our experiment, when the synthetic noise rate reached 10% (which is representative of what can happen when real-world documents are degraded), it was already

Table 4: Unigram overlap, bigram overlap, and simple cosine between extracts and human-created summaries (the numbers in parentheses are the corresponding values between the documents and the original text).

| | Unigram overlap | Bigram overlap | Cosine |
|-----------|-----------------|----------------|-------------|
| Original | 0.85 (1.00) | 0.75 (1.00) | 0.51 (1.00) |
| Snoise.05 | 0.55 (0.61) | 0.38 (0.50) | 0.34 (0.65) |
| Snoise.10 | 0.41 (0.41) | 0.22 (0.27) | 0.25 (0.47) |
| Snoise.15 | 0.25 (0.26) | 0.10 (0.13) | 0.20 (0.31) |
| Snoise.20 | 0.17 (0.19) | 0.04 (0.07) | 0.14 (0.23) |
| Snoise.25 | 0.18 (0.14) | 0.04 (0.04) | 0.09 (0.16) |
| OCR.clean | 0.86 (0.97) | 0.78 (0.96) | 0.50 (0.93) |
| OCR.light | 0.62 (0.63) | 0.47 (0.55) | 0.36 (0.65) |
| OCR.dark | 0.81 (0.70) | 0.73 (0.65) | 0.38 (0.66) |
| OCR.fax | 0.77 (0.84) | 0.67 (0.79) | 0.48 (0.86) |
| OCR.skew | 0.84 (0.97) | 0.74 (0.96) | 0.48 (0.93) |

difficult for a human to recover the information intended to be conveyed from the noisy documents.

Keywords, noun phrases, or headline-style summaries are informative indications of the main topic of a document. For documents with high noise rates, extracting keywords or noun phrases is a more realistic and attainable goal than sentence extraction. Still, it may be desirable to correct the noise in the extracted keywords or phrases. There has been past work on correcting spelling mistakes and errors in OCR output; these techniques would be useful in noisy document summarization.

To choose an appropriate granularity for summary presentation, we need to have an assessment of the noise level in the document. In subsection 4.3, we discuss ways to measure this quantity.

4.2 Using other information sources

In addition to text, target documents contain other types of useful information that could be employed in creating summaries. As noted previously, Chen and Bloomberg’s image-based summarization technique avoids many of the problems we have been discussing by exploiting document layout features. A possible approach to summarizing noisy documents, then, might be to use their method to create an image summary and then apply OCR afterwards to the resulting page. We note, though, that it seems unlikely this would lead to an improvement of the overall OCR results, a problem which may almost certainly must be faced at some point in the process.

4.3 Assessing error rates without ground-truth

The quality of summarization is directly tied to the level of noise in a document. Summarization results are not seriously impacted in the presence of mi-

nor errors, but as errors increase, the summary may range from being difficult to read to incomprehensible.

In this context, it would be useful to develop methods for assessing document noise levels without having access to the ground-truth. Such measurements could be incorporated into summarization algorithms for the purpose of avoiding problematic regions, thereby improving the overall readability of the summary. Past work on attempting to quantify document image quality for predicting OCR accuracy [2, 3, 9] addresses a related problem, but one which exhibits some significant differences.

Intuitively, OCR may create errors that cause the output text to deviate from “normal” text. Therefore, one way of evaluating OCR output, in the absence of the original ground-truth, is to compare its features against features obtained from a large corpus of correct text. Letter trigrams [5] are commonly used to correct spelling and OCR errors [1, 15, 26], and can be applied to evaluate OCR output.

We computed trigram tables (including symbols and punctuation marks) for 10 days of AP news articles and evaluated the documents used in our experiment. As expected, OCR errors create rare or previously unseen trigrams that lead to higher trigram scores in noisy documents. As indicated in Table 5, the ground-truth (original) documents have the lowest average trigram score. These scores provide a relative ranking that reflects the controlled noise levels (Snoise.05 through Snoise.25), as well as certain of the real OCR data (OCR.clean, OCR.dark, and OCR.light).

Different texts have very different baseline trigram scores. The ranges of scores for clean and noisy text overlap. This is because some documents contain more instances of frequent words than others (such as “the”), which bring down the average scores. This

Table 5: Average trigram scores.

| | Trigram score |
|-----------|---------------|
| Original | 2.30 |
| Snoise.05 | 2.75 |
| Snoise.10 | 3.13 |
| Snoise.15 | 3.50 |
| Snoise.20 | 3.81 |
| Snoise.25 | 4.14 |
| OCR.clean | 2.60 |
| OCR.light | 3.11 |
| OCR.dark | 2.98 |
| OCR.fax | 2.55 |
| OCR.skew | 2.40 |

issue makes it impractical to use trigram scores in isolation to judge OCR output.

It may be possible to identify some problems if we scan larger units and incorporate contextual information. For example, a window of three characters is too small to judge whether the symbol @ is used properly: $a@b$ seems to be a potential OCR error, but is acceptable when it appears in an email address such as *lsa@bbb.com*. Increasing the unit size will create sparse data problems, however, which is already an issue for trigrams.

In the future, we plan to experiment with improved methods for identifying problematic regions in OCR text, including using language models and incorporating grammatical patterns. Many linguistic properties can be identified when letter sequences are encoded in broad classes. For example, long consonant strings are rare in English text, while long number strings are legal. These properties can be captured when characters are mapped into carefully selected classes such as symbols, numbers, upper- and lower-case letters, consonants, and vowels. Such mappings effectively reduce complexity, allowing us to sample longer strings to scan for abnormal patterns without running into severe sparse data problems.

Our intention is to establish a robust index that measures whether a given section of text is “summarizable.” This problem is related to the general question of assessing OCR output without ground-truth, but we shift the scope of the problem to ask whether the text is summarizable, rather than how many errors it may contain.

We also note that documents often contain logical components that go beyond basic text. Pages may include photographs and figures, program code, lists, indices, etc. Tables, for example, can be detected, parsed, and reformulated so that it becomes possible to describe their overall structure and even allow

users to query them [12]. Developing appropriate ways of summarizing such material is another topic of interest.

5 Conclusions and Future Work

In this paper, we have discussed some of the challenges in summarizing noisy documents. In particular, we broke down the summarization process into four steps: sentence boundary detection, preprocessing (part-of-speech tagging and syntactic parsing), extraction, and editing. We tested each step on noisy documents and analyzed the errors that arose. We also studied how the quality of summarization is affected by the noise level and the errors made at each stage of processing.

To improve the performance of noisy document summarization, we suggest extracting keywords or phrases rather than full sentences, especially when summarizing documents with high levels of noise. We also propose using other sources of information, such as document layout cues, in combination with text when summarizing noisy documents. In certain cases, it will be important to be able to assess the noise level in a document; we have begun exploring this question as well. Our plans for the future include developing robust techniques to address the issues we have outlined in this paper.

Lastly, we regard presentation and user interaction as a crucial component in real-world summarization systems. Given that noisy documents, and hence their summaries, may contain errors, it is important to find the best ways of displaying such information so that the user may proceed with confidence, knowing that the summary is truly representative of the document(s) in question.

References

- [1] R. Angell, G. Freund, and P. Willet. Automatic spelling correction using a trigram similarity measure. *Information Processing and Management*, 19(4):255–261, 1983.
- [2] L. R. Blando, J. Kanai, and T. A. Nartker. Prediction of OCR accuracy using simple image features. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 319–322, Montréal, Canada, August 1995.
- [3] M. Cannon, J. Hochberg, and P. Kelly. Quality assessment and restoration of typewritten document images. Technical Report LA-UR 99-1233, Los Alamos National Laboratory, 1999.
- [4] F. R. Chen and D. S. Bloomberg. Summarization of imaged documents without OCR.

- Computer Vision and Image Understanding*, 70(3):307–320, 1998.
- [5] K. Church and W. Gale. Probability scoring for spelling correction. *Statistics and Computing*, 1:93–103, 1991.
- [6] Document Understanding Conference (DUC): Workshop on Text Summarization, 2002. <http://tides.nist.gov/>.
- [7] J. Esakov, D. P. Lopresti, and J. S. Sandberg. Classification and distribution of optical character recognition errors. In *Proceedings of Document Recognition I (IS&T/SPIE Electronic Imaging)*, volume 2181, pages 204–216, San Jose, CA, February 1994.
- [8] Y. Gotoh and S. Renals. Sentence boundary detection in broadcast speech transcripts. In *Proceedings of ISCA Tutorial and Research Workshop ASR-2000*, Paris, France, 2000.
- [9] V. Govindaraju and S. N. Srihari. Assessment of image quality to predict readability of documents. In *Proceedings of Document Recognition III (IS&T/SPIE Electronic Imaging)*, volume 2660, pages 333–342, San Jose, CA, January 1996.
- [10] D. Harman and M. Liberman. *TIPSTER Complete*. Linguistic Data Consortium, University of Pennsylvania, 1993. LDC catalog number: LDC93T3A. ISBN: 1-58563-020-9.
- [11] C. Hori and S. Furui. Advances in automatic speech summarization. In *Proceedings of the 7th European Conference on Speech Communication and Technology*, pages 1771–1774, Aalborg, Denmark, 2001.
- [12] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. A system for understanding and reformulating tables. In *Proceedings of the Fourth IAPR International Workshop on Document Analysis Systems*, pages 361–372, Rio de Janeiro, Brazil, December 2000.
- [13] H. Jing. *Cut-and-paste Text Summarization*. PhD thesis, Department of Computer Science, Columbia University, New York, NY, 2001.
- [14] H. Jing, R. Barzilay, K. McKeown, and M. Elhadad. Summarization evaluation methods: experiments and analysis. In *Working Notes of AAAI Symposium on Intelligent Summarization*, Stanford University, CA, March 1998.
- [15] K. Kuckich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, 1992.
- [16] I. Mani. *Automatic Summarization*. John Benjamins Publishing Company, Amsterdam/Philadelphia, 2001.
- [17] M. McCord. *English Slot Grammar*. IBM, 1990.
- [18] D. Miller, S. Boisen, R. Schwartz, R. Stone, and R. Weischedel. Named entity extraction from noisy input: Speech and OCR. In *Proceedings of the 6th Applied Natural Language Processing Conference*, pages 316–324, Seattle, WA, 2000.
- [19] D. Palmer and M. Hearst. Adaptive multilingual sentence boundary disambiguation. *Computational Linguistics*, 23(2):241–267, June 1997.
- [20] D. D. Palmer and M. Ostendorf. Improving information extraction by modeling errors in speech recognizer output. In J. Allan, editor, *Proceedings of the First International Conference on Human Language Technology Research*, 2001.
- [21] J. C. Reyner and A. Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, Washington D.C., 1997.
- [22] M. Riley. Some applications of tree-based modelling to speech and language. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 339–352, Cape Cod, MA, 1989.
- [23] K. Taghva, J. Borsack, and A. Condit. Effects of OCR errors on ranking and feedback using the vector space model. *Information Processing and Management*, 32(3):317–327, 1996.
- [24] K. Taghva, J. Borsack, and A. Condit. Evaluation of model-based retrieval effectiveness with OCR text. *ACM Transactions on Information Systems*, 14:64–93, January 1996.
- [25] Translingual Information Detection, Extraction and Summarization (TIDES). <http://www.darpa.mil/iao/tides.htm>.
- [26] E. Zamora, J. Pollock, and A. Zamora. The use of trigram analysis for spelling error detection. *Information Processing and Management*, 17(6):305–316, 1981.

Rough and Degraded Document Interpretation by Perceptual Organization

Eric Saund, David Fleet, James V. Mahoney, and Daniel Larner

Perceptual Document Analysis Area
Palo Alto Research Center
3333 Coyote Hill Rd. Palo Alto, CA 94304
{saund, fleet, jvmahon, larner}@parc.com

Abstract

Not all document images of interest are comprised of properly scanned, cleanly printed text in a known language, formatted according to standard layout conventions. Sometimes tools are needed to deal with rough documents. Rough documents include handwritten notes, sketches, drawings, annotations, doodles, specialized notations, unconventional layouts, and poorly imaged markings. Despite their unruly diversity, rough documents embody visual structure which is accessible and exploitable by humans by virtue of our powerful perceptual apparatus. We believe that to achieve breadth and depth in interpreting images of rough documents, computer systems will require a foundation of image analysis approximating the human visual stage postulated as Perceptual Organization. This whitepaper motivates and outlines a research program along these lines. A viewpoint that raises the identification of visual perceptual structure as a primary objective helps to open a broader range of tasks for document image analysis, beyond character-to-text transcription.

1 Introduction: Motivation

1.1 Rough Documents

To a human observer, Figures 1a and 1b depict more-or-less the same information. While current text and graphics document recognition technology is capable of transcribing 1b into ascii text and graphical shape models, 1a is far out of reach. Yet, many applications are more likely to encounter images resembling Figure 1a, that is, arising from unpredictable sources and uncontrolled imaging conditions. We may refer to documents whose image content extends beyond properly scanned, cleanly printed text in a known language, formatted according to standard layout conventions, as *rough documents*. Rough documents include handwritten notes, sketches, drawings, annotations, doodles, specialized notations, unconventional layouts, and poorly imaged markings. Figure 2 presents more

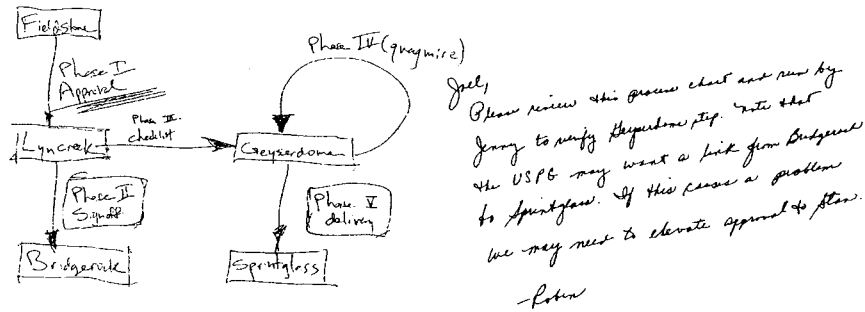
examples of rough documents whose interpretation by machines is far beyond current technology.

The difficulties posed by rough documents are not just a matter of image degradation due to fading ink, multiple-generation copies, or poor quality scanning. Certainly imaging quality can be a factor, as shown in Figures 2a, 2b, and 2f, which were probably captured with digital cameras. But more significantly, rough documents are typically characterized by unconstrained semantic content rendered in idiosyncratic styles, by casual and imprecise marking processes. The result is tremendous unpredictability in what textual and graphical material needs to be recognized, and tremendous variability in how this material appears in a document image.

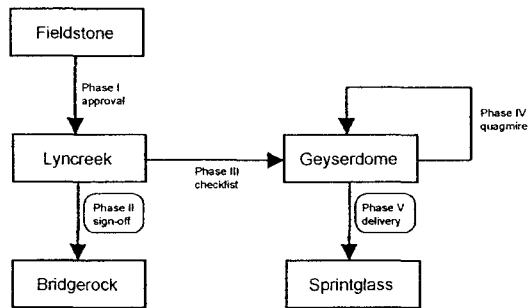
Despite their unruly diversity, rough documents nonetheless embody visual structure which is accessible and exploitable by humans by virtue of our powerful perceptual apparatus. At a glance we identify not only lines and columns of written material, but also printed or hand drawn underlines, encirclings, arrows, figures, the distinct slants of annotations, notable features arising from logos or rubber stamps, coffee stains, graphical arrangements, and so forth. Content elements of this sort are found across document domains. They are all ingredients of comprehensive interpretations of images, and they contribute to critical base-level representations of the layout and coarse-level content of a document necessary to orient and direct localized OCR/ICR, symbol classification, and graphics recognition procedures. We believe that significant machine interpretation of rough documents will require, at a foundational level, algorithms and representations capable of identifying and labeling visual structure comparable to human perceptual processes.

1.2 Tasks

Because of their potentially unpredictable and mixed content, computer systems for managing rough documents may be required to support tasks other than



a



Joel,

Please review this process chart and run by Jenny to verify Geyserdome step. Note that the USPG may want a link from Bridgerock to Sprintglass. If this causes a problem we may need to elevate approval to Stan.

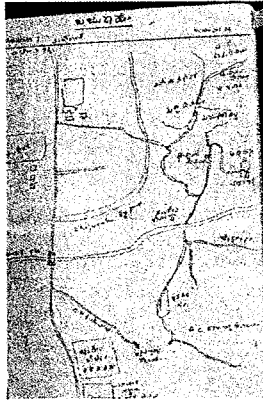
-Robin

b

Figure 1: Rough and formal versions of mixed text and graphics content.

direct transcription. Even when classical recognition cannot succeed, for example in transcribing sloppy handwriting or an unfamiliar language, image analysis tools can still be useful for other purposes. Examples include:

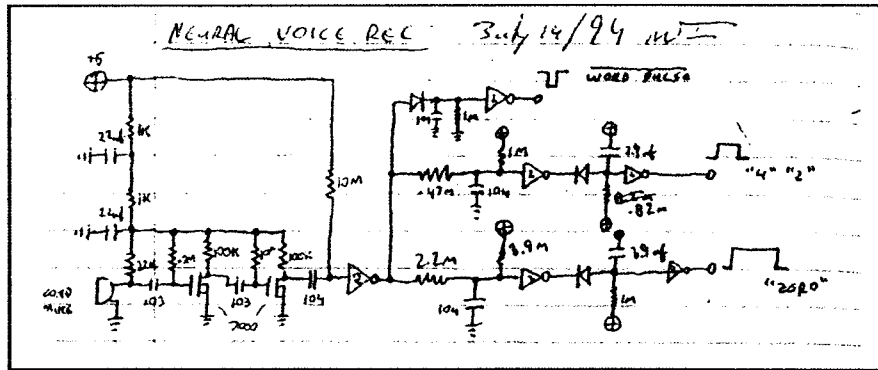
- *Sorting* and classifying documents according to genre or other properties. For example, without attempting to actually transcribe the circuit, the visual qualities of Figure 2d indicate that it is a schematic diagram, it is hand-drawn, it is drawn on notebook paper, and it contains some titular text. By contrast, again prior to performing any text recognition, 2g has the visual characteristics of a receipt, with logos and a watermark.
- *Excerpting* significant sections of rough documents, for example separating the scattered handwritten text of Figure 2e from the photographs, to be entered into a handwriting database. In Figure 2d, perceptual-level recognition of the parallel line structure is critical to the lifting of the writing and drawing from the background rule lines of the paper.
- *Annotating* and *cross-linking* in an electronic document management system would benefit from the ability to attach annotations and links to particular perceptually salient image items, which in turn should be available as instantiated image objects to the underlying system. Even more beneficial would be the ability to detect and flag potentially interesting visual events occurring on rough documents on the basis of their perceptual qualities, such as logos, stains, watermarks, stamps, fingerprints, signatures, and sketches.
- *Editing* image material in rough documents offers the ability to rapidly assemble new documents by re-using complex text and graphics found on existing documents, without having to convert to structured formats. For example, using the perceptually-supported document image editor described in Section 4, the xxxx.
- *Transcribing* handwritten text and diagrams to ascii and structured graphics formats requires the ability to follow the flow of the text regardless of slant and uneven flow, and to recognize graphical structures despite their failure to obey



a

b

c



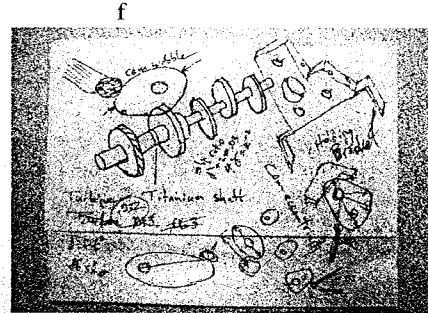
d

WORK
I WENT TO WORK ON THE ASSEMBLY LINE AT CHEVROLET. IT WAS HEAVY!

TRAINING
WHILE WORKING AT CHEVROLET I KEPT DRAWING IN MY SPARE TIME. HAVING A RELATIVE THAT WAS A COMMERCIAL ARTIST, I WOULD SHOW HIM MY DRAWINGS AND HE WOULD CRITIQUE THEM. EVENTUALLY I HAD A PROPOSAL THAT I THOUGHT WAS BETTER. (IT WASN'T) AND WENT OUT TO FIND A NEW JOB AS AN ARTIST.

MY STUDIO
I LET MY FIRST ART TOILET TO BE AT A SMALL APARTMENT BUILDING. FOR NOW, THE STUDIO I WAS OPENING AT CHAVILLE. LIFE IS GOOD!

e



f

ZERRAAYA - METN
TEL or FAX: (04) 280161
زرعيا - مطن
تلون وفاكس: (٠٤) ٢٨٠١٦١

Happy Childhood Home
بيت الطفولة
السعيدة

LL [redacted] ل ل
Date 11/1/2000 التاريخ

Poçu de LEBANESE CLUB of CHICAGO وصلتان
La somme de quatre mille dollars U S مائة
Pour Happy childhood Home ذلك

N° 0034

g

RADIO: NEWS

Compte rendu de Dr. Boris Barthelemy, directeur de l'Institut de Recherche sur les Radiosondes de l'Armée Française.

Il existe une très grande variété de radiosondes. Elles sont utilisées pour la mesure de la température, de la pression, de l'humidité, de la vitesse du vent, de la direction du vent, de la hauteur, de la densité de l'air, de la conductivité électrique, de la fréquence des ondes radio, etc.

Les radiosondes sont classées en deux catégories: les radiosondes météorologiques et les radiosondes militaires.

Les radiosondes météorologiques sont utilisées pour la mesure de la température, de la pression, de l'humidité, de la vitesse du vent, de la direction du vent, de la hauteur, de la densité de l'air, de la conductivité électrique, de la fréquence des ondes radio, etc.

Les radiosondes militaires sont utilisées pour la mesure de la température, de la pression, de l'humidité, de la vitesse du vent, de la direction du vent, de la hauteur, de la densité de l'air, de la conductivité électrique, de la fréquence des ondes radio, etc.

h

Figure 2: Rough documents arise from unpredictable sources and uncontrolled imaging conditions.

strict graphical drafting rules.

- *Indexing and Retrieval* of rough documents would benefit greatly from automated means for labeling image content on the basis of visual qualities, even short of formal recognition in terms of natural language or graphical language constructs.

We are confident that exposure to practical application scenarios for document image management will lead to articulation and refinement of these non-traditional tasks supported by perceptual level image analysis of rough documents, and the invention of new ones.

2 Design Principles

2.1 Strong and Weak Models

Conceptualizations of both natural and artificial visual systems tend to be organized around a sequence of processing stages. The kind of circuits necessary at the sensory levels to manage dynamic range and contrast variation are different from the algorithms employed for, say, feature detection, which in turn are different from the computations involved in classification, model matching, and search.

Technology for recognition of document images bifurcates at an early stage between processing steps employed for text recognition and those employed for graphics recognition, respectively. These may have in common an image processing stage that may include thresholding, followed by a stage performing text/graphics separation. Then, text recognition processes typically follow the stages of skew detection, page layout analysis, word and character segmentation, and character symbol classification. Wherever possible, lexical or other semantic constraints are used to resolve ambiguous classifications and improve accuracy. Graphics recognition processes typically follow the stages of graphics vectorization, symbol matching, and knowledge-driven parsing of the vectors in terms of domain models particular to the drawing domain [Tombre and Chhabra, 98]. Examples of domain models include drafting rules for drawing mechanical parts and their dimensioning information, the syntactic rules governing the components and wires of electrical schematics, and architectural conventions for arranging walls, doors, windows, and furniture.

Both text and graphics lines of processing rely heavily on *strong prior models* of document image content to resolve ambiguities, to constrain possible interpretations, and to delimit and guide successive processing steps. These assumptions are appropriate when applied to documents whose image content is in fact confined, for example to pages of printed

text or scans of engineering drawings. But the larger document image domain that includes rough documents cannot be approached in this way. Human perception is capable of detecting and making use of visual structure that violates standard norms defined for proper text layout and graphical drawing rules. Machine vision systems for rough documents must have this flexibility as well.

2.2 Knowledge Levels

Such a perspective leads our research group to a five-level framework for document image analysis that modularizes knowledge in a slightly different way. This is illustrated in Figure 3.

1. Early Processing: Early visual processing encompasses image processing and initial stages of feature detection. Electronic images of documents captured not from scanners but from digital camera images of notebooks, whiteboards, flipcharts, projected images, etc. will in general carry artifacts of the imaging process due to sensor noise, sampling, focus issues, geometric distortion, dirt, smudges, illumination color and intensity gradients. These cause image degradations that impede extraction of the truly significant markings. One function of early processing is to “clean up” or condition the input using various image processing techniques such as color normalization, contrast equalization, adaptive thresholding, despeckling, etc. A second function of early processing is to identify primitive visual features such as lightness intensity edges, ridges, and solid regions.

2. Perceptual Organization: The Gestalt psychologists of the early 20th century identified a number of principles that characterize spatial structure the human visual system is adept at detecting. [Wertheimer, 23, Kanizsa 79, Green, 00]. Among the most important of these “Gestalt laws” are curvilinear alignment, or smoothness, closure, spatial proximity, feature similarity, and symmetry. These properties are viewed as being important clues to object cohesiveness and identity in natural scenes, and they are found to strongly govern the perception of synthetic images as well.

Because documents as created by people are intended to be seen and interpreted by our visual systems, it is likely that their structure naturally tends to reflect the fundamental visual capacities that we are endowed with. Indeed, Figure 4 illustrates Gestalt laws by way of positive and negative examples that respectively validate and violate our automatic perceptions. One rarely encounters a document whose semantic intent is depicted spatially in violation of the principles of perceptual organization. And when we do, these documents are confusing and

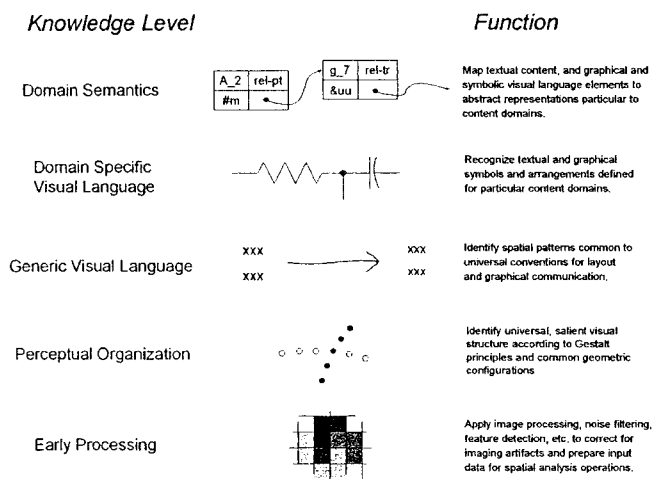


Figure 3: A Research framework of five levels of analysis for rough documents.

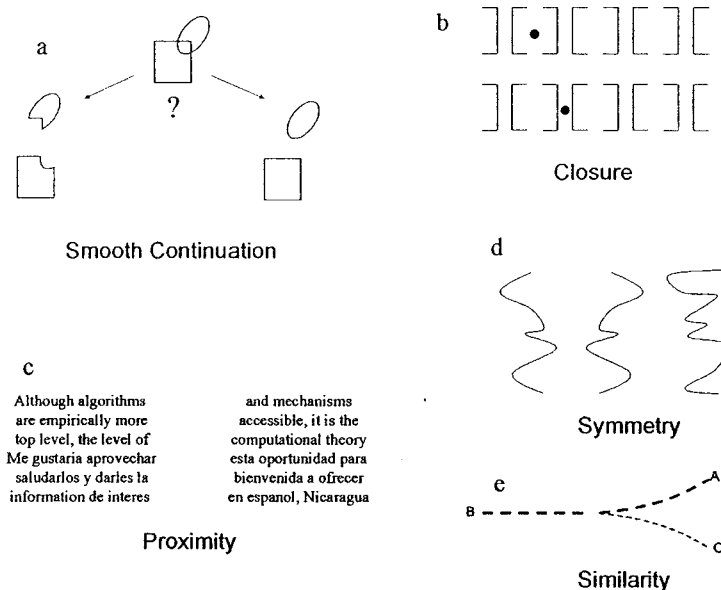


Figure 4: Illustrations of five Gestalt principles of visual perceptual organization. a. The central figure appears to be a combination of parts having smooth boundaries. b. The top dot appears to fall on a foreground object defined by a nearly closed boundary contour, while the bottom dot appears to lie on the background. c. The apparent visual partitioning of the text based on proximity overrides a partitioning based on semantic content. d. The curves that appear to go together are the pair forming a bilateral symmetry. e. The curve appears to continue from point B on the basis of similarity of its local properties.

difficult to read.

We believe that visual Perceptual Organization (PO) has an important role to play in computational models for rough documents. Starting with an undifferentiated array of pixels, the purpose of this stage is to identify chunks, groupings, and patterns in image material reflecting visually salient structures that are likely to reflect syntactically and semantically meaningful elements and relationships. Short of committing to conclusive mappings to the document's semantic domain, PO provides a rich set of candidates and building blocks to draw upon in constructing these mappings. Moreover, the knowledge required to do this processing, i.e. the data structures and algorithms required to in essence "implement" the Gestalt laws, would apply universally, across all document content domains, spanning both text and graphics.

The principles of PO. are supported by many extremely compelling psychophysical demonstrations, but have proven very difficult to formulate as generally applicable computational algorithms. This remains a subject of active research in Computational Vision [Witkin and Tennenbaum, 83; Sarkar and Boyer, 94; Boyer and Sarkar, 99; Boyer and Sarkar, 00]; Jacobs and Lindenbaum, 01].

3. Generic Visual Language: In addition to implicitly respecting the principles of human perceptual organization as it supports visual interpretation of natural scenes, we believe that human graphic communication embodies another level of structure consisting of standards and conventions that are culturally evolved and culturally acquired. In our hypothesis, this *Generic Visual Language* (GVL), serves certain universally important functions in communication such as demarking distinct groups, indicating progressions or other relationships, labeling, referring, and so forth. Some conventions of GVL involve graphical elements or symbols, such as lines, arrows, and encirclings, while others pertain to spatial arrangements, such as paragraph structure, titles and captioning, tabular structure, hierarchical indentations, and the proximity of text labels to graphics objects they refer to. Examples are shown in Figure 5.

The notion of Generic Visual Language, and a detailed account of its constituent patterns and rules, has not been a focus of study in the literature, although a few researchers have examined aspects of this topic [Bowman, 68; Bertin, 83; Tversky, 95; Tversky, 00; Tversky et al, 00]. We anticipate that the challenges of characterizing the range, diversity, and universality of Generic Visual Language will prove a rich area of investigation for the diagrams community. As with Perceptual Organization, this research must span a range of disciplines and activi-

ties, including gathering and cataloging of data, experimental studies with diagram users, development and implementation of algorithms for recognizing GVL constructs in representations of images, building cognitive models for how spatial and symbolic elements contribute to reading of diagrams, etc.

4. Domain-Specific Visual Language: Whether or not universal conventions for spatial communication play a significant role in interpreting documents, it is clear that particular domains of discourse have developed their own specialized vocabularies and symbologies. Consisting of specialized symbols, linework, layout templates, and syntaxes for arranging these, *domain-specific visual languages* (DSVLs) are numerous and varied. We assume that all Domain-Specific Visual Languages respect the principles of Perceptual Organization and that PO-based segmentation and grouping will contribute to computational models for interpreting documents in terms of these DSVLs. Generic Visual Language conventions may provide a foundation and a resource for specialization in specific domains. For example, labeled arrows (lines with arrowhead symbols) may indicate a directed relationship between "from" and "to" object nodes, but particular domains may enforce rules such as the number of arrows allowed to enter or exit nodes, whether arrows may cross one another, and so forth.

5. Domain Semantics: While documents themselves are by nature spatially constructed representations renderable as images, the concepts and relationships they refer to may not be. Therefore, we assume that document interpretation systems may ultimately interact with knowledge and procedures at a non-spatial level, which we may call *domain semantics*. Some of this apparatus may be devoted to mapping between representations at this level and the domain-specific visual language level of description, while other machinery may be purely abstract or propositional and have nothing to do with document images at all.

Within this five-stage hierarchy, the Perceptual Organization and Generic Visual Language stages afford a ripe opportunity to develop *weak prior models* for document image analysis, that is, to identify meaningful visual structure without resorting to strong assumptions about what will be found. The proposed emphasis is on building tools that will deliver at least *some* level of meaningful interpretation across a wide range of document image input, as opposed to performing very well a narrow class of images but failing on any input that falls outside the class. Such a foundation is likely to support not only the interpretation of rough documents, but formal documents as well, especially when they contain

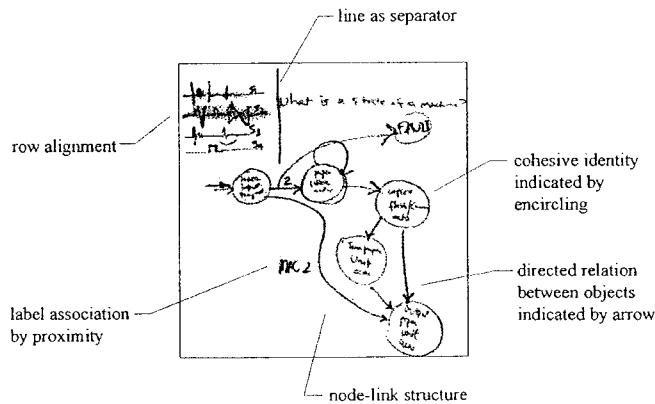


Figure 5: Examples of Generic Visual Language: visual patterns and notational conventions that convey formal or informal relationships, found across content domains.

rough aspects, such as handwritten annotations, logos, tables, and figures.

3 Technology and Research

Our group has engaged a number of topics focusing on the three earliest knowledge levels in this framework. Our efforts include:

- Image Processing
 - mosaicing of multiple snapshots for a video camera-based whiteboard scanner
 - page image dewarping for a face-up book scanner
 - color normalization for documents imaged with a camera
- Perceptual Organization
 - curve saliency and grouping
 - sketch recognition
 - occluding surface labeling
 - closed path detection
- Visual Language Recognition
 - structural page description and matching
 - structural recognition by constraint-based subgraph matching
 - sketch recognition exploiting PO-based data graph rectification
- Applications
 - perceptually-supported digital ink sketch editing

- diagrammatic user interfaces
- physical/virtual collaboration surfaces
- perceptually-supported bitmap-based document image editing

In the following sections we touch on a few of these projects.

3.1 Image Processing

The advent of digital cameras increasingly enables casual document image capture. The resulting images suffer from artifacts not encountered through traditional scanning. One important image processing function is to correct for uneven illumination of the document. Our algorithm for lightness correction is based on inverting the image equation relating illumination, I , surface reflectance R , observed lightness L ,

$$L = IR$$

If one is able to construct an estimate of the illumination I , then the underlying document reflectance can be recovered. A standard approach attempts to recover illumination under the assumption that most of the document consists of background white, on which dark markings are sparsely distributed. Under this model, foreground markings can be detected by high-pass filtering. The lightness values of the background can be interpolated across the resulting foreground mask.

However, this approach fails when the document contains large regions of foreground markings, such as contained in business graphics and artwork. Figure 6b illustrates the results of the commercial pro-

gram, Whiteboard Photo. The fundamental problem is that, because illumination may vary across the image, it is impossible to determine locally whether the RGB value of any given patch is due to colored illumination off a white background surface or reflectance of a bright illuminant off a marked foreground surface.

To address this problem we have recently developed enhanced methods for detecting larger foreground regions, leading to improved color-normalization as exemplified in Figure 6c. We believe that much research remains to be done toward global spatial integration of image cues about lightness and reflectance changes which will lead to a comprehensive solution to this problem.

3.2 Perceptual Organization

The field of computer vision has seen some progress in formalizing middle-level visual processes of Perceptual Organization as computer vision algorithms for such processes as texture segmentation [Malik et al, 99], curvilinear line aggregation [Amir and Lindenbaum, 98], shape decomposition [Johannes et al, 01], and closed path detection [Jacobs, 96], among others. Because of limited efficacy, suitability of image content, and computational cost, rather little of this research has been carried over directly to document image analysis. However, the overall motivation and certain algorithmic techniques for finding perceptual structure in document images has taken root in the document image analysis community. Examples include grouping approaches to page segmentation [Kise, 98], grouping of curvilinear strokes in maps [Thomson and Brooks, 02], and grouping of text elements in engineering drawings [Kasturi et al, 92, Burge, 03].

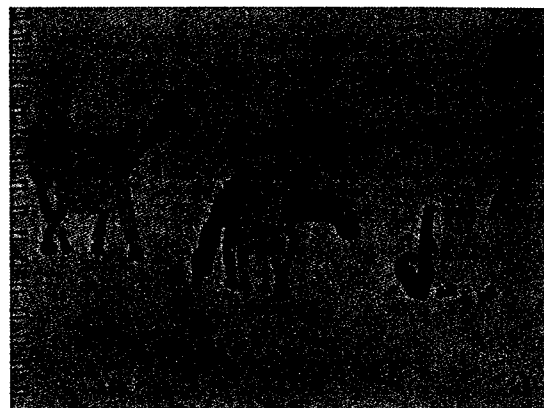
One core paradigm is a three-stage process consisting of the following steps:

- 1 *Segment* the image into primitive elements.
- 2 Form *link* relations among related elements.
- 3 Perform *search* for collections of linked elements that satisfy target criteria.

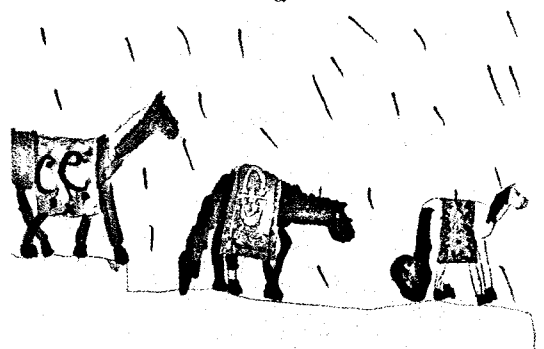
Different kinds of visual structure are detected according to specific criteria, parameters, and algorithms employed for each of the steps.

3.2.1 Segmentation

In document image analysis, the segmentation step almost always starts with the extraction of connected components in a binarized image. In clean images of formal documents, individual characters are frequently segmented into distinct components. Line drawings, on the other hand, undergo a further step of vectorization, which delivers units corresponding to straight or curved segments. Very little



a



b



c

Figure 6: a. Original digital camera image. b. Processing by the commercial product, Whiteboard Photo. Note degradation of the solid color regions. c. Result of our algorithm which detects large foreground regions.

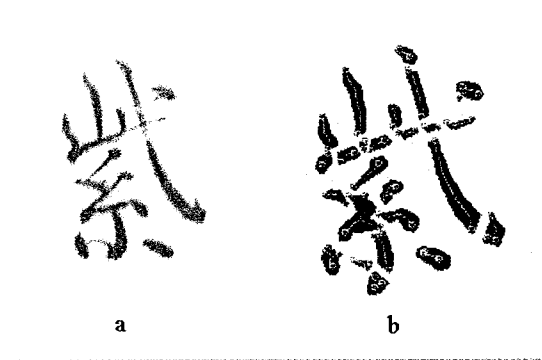


Figure 7: a. One of the ideographs extracted from Figure 2b. b. Exploded view of elements resulting from splitting to extract salient curvilinear fragments.

work has been devoted to articulated segmentation of solid graphic regions, although some commercial systems include detection of photographic regions which are extracted as distinct image objects.

Traditional analysis of degraded documents addresses the problem of distinct semantic image objects touching one another and thereby being inseparable on the basis of connected component segmentation. For text, special algorithms have been developed to perform character segmentation, while some limited amount of work has been done on separating touching symbols and line-art in technical drawings [Shimotsuji, et al, 94].

The problem of separating touching objects is especially important when addressing rough documents, which frequently include hand-drawn sketches and handwritten notes. Our group has focused some attention on this problem, and our approach has evolved over time, starting from one based on iterative classification of connected components and splitting off of curvilinear fragments of limited thickness [Saund, 02]. More recently, we have developed techniques that combine information derived from contour features with scale-invariant detection of “strokes” in order to split off significant curvilinear objects at perceptually natural cut points. Figure 7 presents an example.

3.2.2 Linking

The purpose of linking is to form a graphical substrate of candidates for grouping collections of elements. Linking of image objects is often done based on spatial proximity (e.g. k-neighbors) or the Delaunay graph. In their simplest form, links are purely categorical, all-or-none. Greater use of links is gained by endowing them with attributes reflecting local spatial properties such as distance and orientation between respective pairs of linked objects.

Our research has pushed this idea further by iteratively refining the set of links according to the local spatial environment. In [Mahoney and Fromherz, 02] for example, the initial sets of linked curve ends are augmented to include the transitive closure of any set of links. In [Saund, 03], linking criteria are adjusted adaptively to deliver sets of links of bounded degree. Both this work, and [Saund, 99], each introduce links with rich type attributes which are in turn attributed by local measures of spatial configuration.

3.2.3 Grouping Search

Under the segmentation/linking/grouping framework, grouping amounts to selection of subgraphs of the link graph. The simplest version of this is the selection of transitive closures, assuming initial links have been pruned down to exclude links between elements that should not be grouped together, leaving “islands” of connected components in the graph. This is equivalent to tracing in one dimension, and coloring in two dimensions. More generally, graph partitioning algorithms [Shi and Malik, 00] or search algorithms are applied to generate sets of linked elements that reflect target global criteria. Thus far, these criteria mainly reflect spatial proximity, curvilinear alignment, and path closure. Our group has developed versions of all of these grouping criteria. For example, Figure 8 shows a result from our recent work on the identification of perceptually closed paths in line drawings [Saund, 03]. This technique involves two primary innovations. First, we identified criteria relating both the local progression of lines through junctions, and the global shape of paths, to paths’ perceptual salience. Second, we showed that a bi-directional search procedure ameliorates garden path searches and permits qualified paths in the link graph to be found efficiently.

In general we believe that the issues of forming and refining link graphs, and formulating search criteria and search algorithms to identify perceptually salient visual structure, will continue to be key areas of research important to sensible machine interpretation of rough documents.

3.3 Visual Language Recognition

Perceptually salient collections of primitives labeled as groups at the PO stage represent resources, or building blocks, with respect to which recognition procedures may perform matching to structural models. Under this framework, much latitude remains in the content of a structural model database, and in the matching procedures used.

Under the five knowledge levels identified in Section 2.2, we anticipate that many if not most rough documents will reflect general graphical conventions

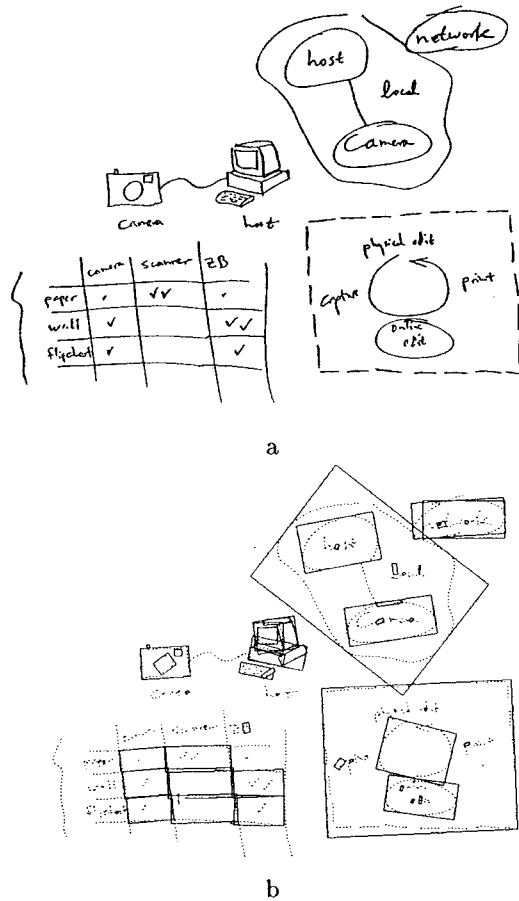


Figure 8: a. Sketched figure. b. Closed paths found by our bidirectional search algorithm.

that may eventually be expressed as a relatively stable database of generic visual language constructs. Elements of generic visual language include the various roles that straight lines, arcs, encirclings, arrows, and proximal text play as communicative devices for grouping, separating, labeling, referring, indicating progression, indicating logical structure, indicating text flow, and so forth.

By contrast, model databases for domain specific visual languages pertain to specific document domains such as journal layouts, schematic diagrams, tables, charts, schedules, technical drawings, and mathematical notations. These databases may be expected to grow and adapt to different application domains, which may nonetheless span both rough and formal expressions of underlying semantic content.

Our research adopts the strategy of seeking a common structural matching framework which will apply to both Generic and Domain Specific visual language.

3.3.1 Structural Modeling and Matching

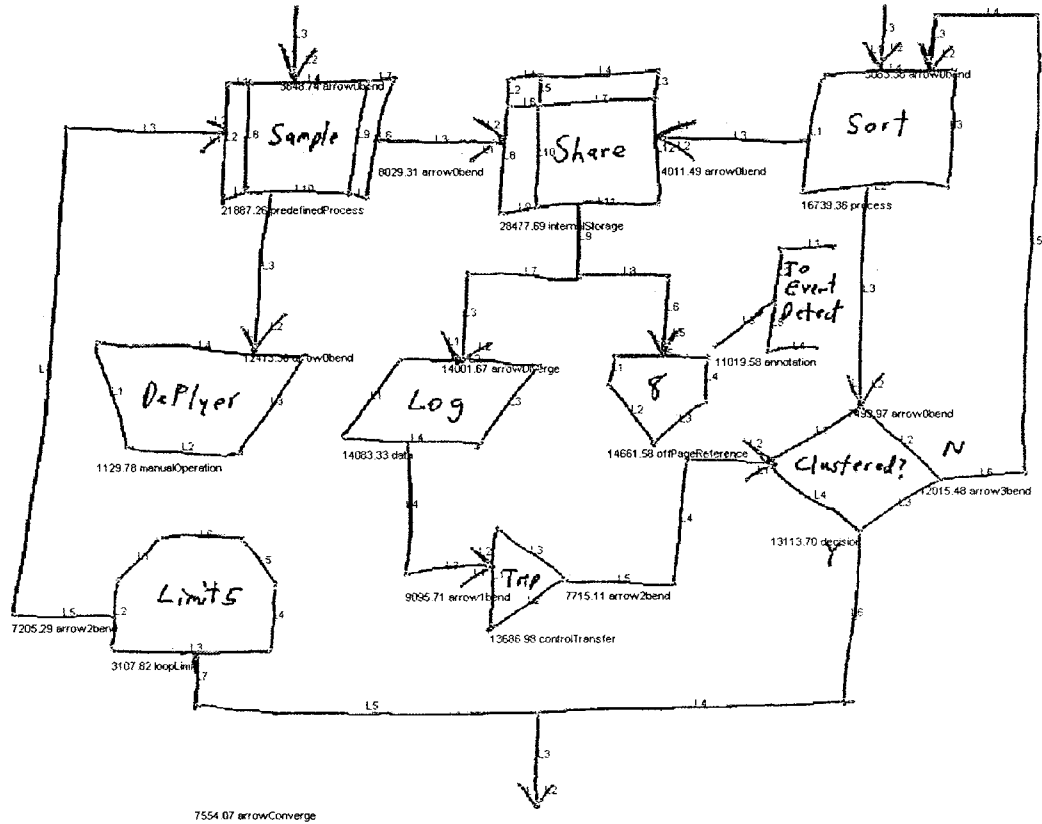
Structural matching recognizes and parses graphical configurations. By "parsing" we mean individually associating each input mark with the model part that it depicts. Because of ambiguity in the association between model objects and data objects, this process must cope with the multiplicity of alternative local groupings produced by the Perceptual Organization stage, and it must be capable of producing alternative plausible interpretations of the same data, ranked by their plausibility. It must cope with the inherent exponential worst-case complexity of structural matching in the graphical domain, and it must be easily extensible with new configuration models.

Our group is developing a subgraph matching formulation, implemented in a constrained optimization framework. See Figure 9. The PO stage generates multiple plausible groupings leading to alternative hypotheses for assignment of model parts to observed data. One set of constraints enforces a unique selection among competing alternatives. The modeling language for specifying configurations allows topological and geometric constraints, both hard and soft, on the spatial relationships among parts that define model objects. These constraints feed in a straightforward way into an overall measure of match quality that serves as the objective function for optimization. We believe that this approach will elegantly handle issues of ambiguity, and provide a solid foundation for dealing with the issues of complexity and open-endedness.

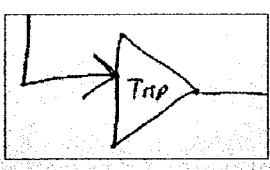
4 Application Platform: Perceptually Supported Image Editing

We are pursuing these lines of research in Perceptual Organization and Visual Language Recognition in rough documents against an application platform that simultaneously tests, and delivers value from, our research as it progresses. This platform, implemented in Java, presents a toolkit for building graphical user interfaces to applications calling upon our research code base.

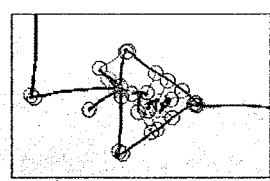
One application, called *ScanScribe*, is a perceptually supported document image editor. At its basic level, *ScanScribe* offers selected functionality found in photographic image editors like Photoshop, but with an interaction model targeted especially to easy selection and manipulation of image material in documents. In particular, color normalization processing described in Section 3.1 performs automatic foreground/background layer separation, in which the background is rendered transparent. Then, as the



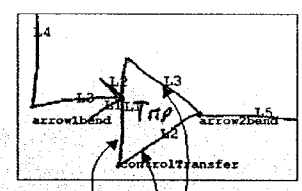
a



b



c



d

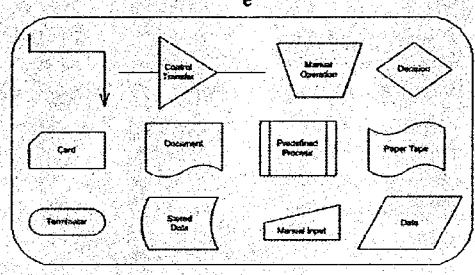


Figure 9: a. Hand-drawn diagram tagged with labels indicating matched parts. b. One item from this diagram. c. Spatial relations among features. d. Corresponding parts found by matching to one item in the model database, e.

user selects and manipulates image objects present in a single input bitmap, they retain their identity as independent bitmap fragments which can be dragged, duplicated, grouped with one another, and carved into yet smaller pieces. By design, this process occurs automatically and below the user's conscious awareness.

While it is quite novel and useful at its base level, ScanScribe presents a platform for testing and validating image analysis research. The ScanScribe architecture is designed to make available the results of any grouping processes that automatically recognize visual structure—at whatever level—in the image. For example, if a recognizer were available that recognized tabular structure, ScanScribe could present the user with easy ways to access, at will, the rows, columns, individual elements, and entire table. Because a user-interactive image editor puts the user intimately in the loop, even partial or imperfect recognition processes can become useful. This is especially significant when working with rough documents, many exemplars of which will prove extremely difficult for machines to interpret without errors for quite some time to come.

Because image editing is a broadly useful functionality, we anticipate that the ScanScribe document image editor may serve as a springboard or base level upon which other, more specialized applications, will be built. For example, a scenario in which a user wishes to perform database search on the ideographs in Figure 2b, requires the selection of this target from the other material in the image which could misdirect any database search processes.

5 Conclusion

Success in document image analysis has to date largely relied on having strong prior models of the input data, including strong models for scanner degradation processes, page layout, font, text content, engineering drawing rules, and so forth. These strong models are inappropriate for rough documents such as handwritten sketches and notes, graphical figures, annotated documents, complex layouts, and casually imaged documents. Instead, we advocate the development of core visual analysis competency at the levels of Perceptual Organization, dovetailing with Image Processing below and Visual Language Recognition above. The visual structure brought forth at these stages will support not only transcription, but will open a broad range of document image processing, analysis, interpretation, and management tasks.

6 References

Amir, A., and Lindenbaum, M., "A Generic Grouping Algorithm and its Quantitative Analysis," *IEEE*

TPAMI, 20:2. 1998.

Bertin, J. *Semiology of Graphics: Diagrams, Networks, Maps*. Univ. of Wisconsin Press, 1983.

Bowman, W.J., *Graphic Communication*. J. Wiley, New York. 1968.

Boyer, K., and Sarkar, S., eds. *Perceptual Organization for Artificial Vision Systems*. Kluwer. 2000.

Burge. M., "Document Image Analysis: Vectorization, Segmentation, Classification & Grouping," "<http://www.computing.armstrong.edu/FanNStaff/burge/java/grouping/grouping.html>"

Green, C., "Introduction to: 'Perception: An introduction to the Gestalt-theorie' by Kurt Koffka (1922)", "<http://psychclassics.yorku.ca/Koffka/Perception/intro.htm>". 2000.

Kanizsa, G., *Organization in Vision: Essays on Gestalt Perception*. Praeger, New York. 1979.

Jacobs, D., "Robust and Efficient Detection of Salient Convex Groups", *IEEE TPAMI*, 18:1. 1996.

Jacobs, D., and Lindenbaum, M., *POCV 2001: The Third Workshop on Perceptual Organization in Computer Vision*, CIS Report CIS-2001-05, Technion, Haifa, Israel. 2001.

Johannes, M., Sebastian, T, Tek, H., and Kimia, B., "Perceptual Organization as Object Recognition Divided by Two," in Jacobs, D., and Lindenbaum, M., *POCV 2001: The Third Workshop on Perceptual Organization in Computer Vision*, CIS Report CIS-2001-05, Technion, Haifa, Israel. 2001.

Kasturi, R, Raman, R., Chennubhotla, C., and O'Gorman, L., "An Overview of Techniques for Graphics Recognition", in Baird, H.S., Bunke, H., and Yamamoto, K., eds., *Structured Document Image Analysis*, Springer-Verlag, Berlin. 1992.

Kise, K., Sato, A., and Iwata, M., "Segmentation of Page Images Using the Area Voronoi Diagram," *CVIU* 70:3. 1998.

Mahoney, J., and Fromherz, M., "Interpreting Sloppy Stick Figures By Graph Rectification," in D. Blostein and Y-B. Kwan, eds., *Graphics Recognition: Algorithms and Applications*, 4th International Workshop, GREC 2001, Springer LNCS 2390., 2002.

Malik, J., Belongie, S., Leung, T., and Shi, J. "Contour and Texture Analysis for Image Segmentation," in K. Boyer and S. Sarkar, *Perceptual Organization for Artificial Vision Systems*, Kluwer. 2000.

Sarkar, S., and Boyer, K., "A Computational Structure for Preattentive Perceptual Organization: Graphical Enumeration and Voting Methods". *SMC(24)*. 1994.

Saund, E., "Perceptual Organization of Occluding Contours of Opaque Surfaces", *CVIU* 76:1. 1999.

Saund, E., "Method and apparatus for separating document image object types", U.S.P. 6,411,733. 2002.

Saund, E., "Finding Perceptually Closed Paths in Sketches and Drawings," *IEEE TPAMI*, in press.

Shi, J., and Malik, J., "Normalized Cuts and Image Segmentation," *IEEE TPAMI*, 22:8. 2000

Shimotsuji, S., Hori, O., and Asano, M., "Robust Drawing Recognition Based on Model-Guided Segmentation", in A. L. Spitz and A. Dengel, eds., *Proc. IAPR Workshop on Document Analysis Systems*, 1994.

Thomson, R., and Brooks, R., "Exploiting Perceptual Grouping for Map Analysis, Understanding and Generalization: The Case of Road and River Networks" in D. Blostein and Y-B. Kwan, eds., *Graphics Recognition: Algorithms and Applications*, 4th International Workshop, GREC 2001, Springer LNCS 2390., 2002.

Tombre, K. and Chhabra, A.K., (Eds). *Graphics Recognition - Algorithms and Systems*. Lecture Notes in Computer Science, vol. 1389. Springer Verlag. 1998.

Tversky, B., Zacks, J., Lee, P., and Heiser, J., "Lines, Blobs, Crosses, and Arrows: Diagrammatic Communication with Schematic Figures". In M. Anderson, P. Cheng, and V. Haarslev, eds. *Theory and Application of Diagrams*, LNCS 1998, Springer. 2000.

Tversky, B.: "Some Ways that Maps and Diagrams Communicate". *Spatial Cognition*. 2000.

Wertheimer, M.. "Laws of Organization in Perceptual Forms", In Ellis, W., ed, *A source book of Gestalt psychology*, Routledge & Kegan Paul, London. 1938 (1923).

A.P. Witkin and J.M. Tenenbaum. On the role of structure in vision. In J. Beck, B. Hope, and A. Rosenfeld. (eds.), *Human and Machine Vision*. Academic Press. 1983.

OCR Accuracy Prediction as a Script Identification Problem

Vitaly Ablavsky, Joshua Pollak, Magnús Snorrason, and
Mark R. Stevens

Charles River Analytics Inc.
625 Mount Auburn St, Cambridge MA 02138

Abstract

Optical Character Recognition (OCR) accuracy prediction and the associated problem of choosing an optimal OCR to process a given document page are two very important issues that need to be addressed when processing degraded documents. In this paper we tackle this problem in the context of an intelligent document enhancement framework that we are developing. By taking a look at the overall system architecture we realize that existing modules may be used more than once in the overall pipeline, performing a different function each time. Specifically, we ask whether a script identification engine, after being suitably trained, can distinguish not between different script types, but between different degradation levels of the same script. We then craft a rules-based classifier to send the document to the most suitable OCR. In our experimental setup we apply our algorithmic framework to a common set of artifacts—broken and merged characters—and employ two OCR engines with different performance characteristics. Our results demonstrate a robust ability to discriminate between clean, broken-character, and merged-character documents. Finally, we demonstrate that using our system to choose between the two OCR engines produces a better overall accuracy than either engine can produce on its own.

1 Introduction

Document image enhancement is often a necessary preprocessing step in the overall OCR pipeline. Without enhancement, the output of an OCR engine on a given document may be of low accuracy. Several recent research efforts have attempted to predict OCR accuracy for a given document image [1], [2]. At the same time, work on document image enhancement has resulted in promising approaches that boost OCR performance [3], [4], [5].

In this paper we explore two themes. The first theme is a system view that unifies document image enhancement and OCR prediction in the context of our Autonomous Intelligent Document Analysis (AIDA)

system. This part of the paper is speculative since the system is still under development. We then ask the question whether some modules of AIDA can serve more than one purpose in the data flow. In particular, we wish to find out whether a script identification engine [6], [7], [8], [9], [10] can be retrained to recognize different degradation levels of the same script (e.g. Latin). In the remainder of the paper we describe our accuracy prediction approach, the experimental setup used in validation, and the results produced.

This paper is organized as follows. Section 2 describes a system architecture view of our document enhancement pipeline. In section 3 we describe a script-identification based approach to detecting merged text and a separate approach for detecting broken characters. The experimental setup is discussed in section 4, followed by rule-based OCR selection in section 5. The following section presents validation results, and the last section summarizes the work.

2 A System View of Document Enhancement

Consider the diagram in Figure 1. This diagram represents the most straightforward document enhancement pipeline. The document enhancement module is viewed as a black box.

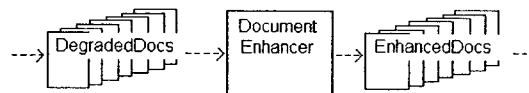


Figure 1: Basic document enhancement data flow

Now consider the pipeline in Figure 2. In this system a separate OCR accuracy prediction module estimates the quality of each document image. The output of the prediction module feeds into some form of an intelligent control which decides on the best action. The action is to present the document image to an OCR system (the diagram shows the case where the choice is between OCR-A and OCR-B) or to a human who will manually transcribe (key in) the document.

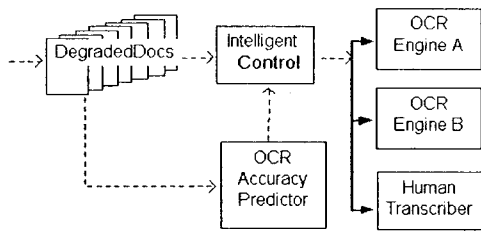


Figure 2: Basic OCR control system

The document enhancement and OCR control pipelines can be combined, as shown in Figure 3.

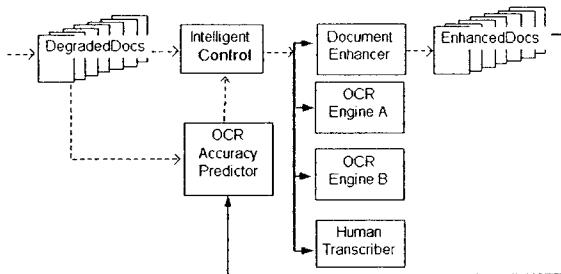


Figure 3: Unifying image enhancement and accuracy prediction

The intelligent control chooses between the same three actions as in the previous case plus the document enhancement. However, simply offering document enhancement as a fourth output is insufficient, which is why an arrow connects *EnhancedDocs* back to the OCR accuracy predictor. First, document enhancement by itself is not a goal, thus the newly generated document needs to re-evaluated by the control module. Second, by putting “feedback” into the system we open up the door to a whole set of online optimization approaches.

The decision on which action to take is best formulated in terms of risk or cost. For the simplest case decision where the choice is just between using an OCR or human transcriber, a typical cost model has this form

$$Cost = \alpha N_{correct} + \beta N_{FA} + \gamma N_{MD} \quad (1)$$

This model expresses the relative cost of either incorrectly labeling a document image as poor (False Alarm) and unnecessarily routing it to the human transcriber, or not detecting that the document is of poor quality (Missed Detection), sending it to an OCR and obtaining very low accuracy. Since correct accuracy prediction is the goal, α is typically set to 0. For a system such as shown in Figure 3 above, such a cost model needs to weigh in each element from a 4x4 confusion matrix representing the 16 possible outcomes of choosing which of four labels (“Enhance”, “OCR-A”, “OCR-B”, or “Human”) to apply to each document.

The control-theoretic approach to the document enhancement pipeline lends itself to exploring novel ways to boost the overall performance. One such approach is to use an existing script identification engine as a key component of the OCR accuracy

predictor. The intuition is that the output of the script identification—which in our case answers the question whether an input document is English or Russian—can be further mined to answer a parallel query. Namely, a script identification algorithm can tell us how “English-like” is the given English document.

The standard script identification problem statement is to distinguish between two (or more) languages. This formulation does not apply in the case of degradation estimation, degraded document images of English origin do not begin to resemble Russian document. A possible formulation is to consider degraded English documents to belong to its own script class, call it English*, and pose the script identification problem as distinguishing between the classes

$$\{\text{English, English}^*\}$$

and to take the raw likelihood of the English label as the predictor of the overall OCR accuracy. The accuracy predictor can either solely rely on script identification to assess the document quality or it can fuse that information with its own quality assessment metrics.

This double use of script identification is sketched out in Figure 4. The two outputs from the *Script Identification* block imply that the script identification engine be run twice for each document (with different parameter settings). First to label an input document according to language, and second as part of the accuracy prediction. The *Intelligent Control* module uses the language label to select a language-specific OCR for a given page image. Then, using the accuracy prediction labels, the control module can make one of three decisions: 1) to further down-select among eligible language-specific OCR’s the one that it estimates would best handle the given page, 2) to send the document to the *Document Enhancer*, or 3) to decide that this document is degraded beyond the capabilities of machine processing and send it to a human transcriber.

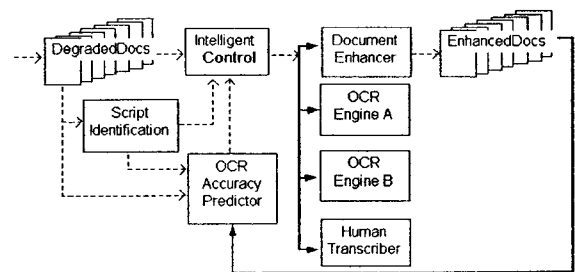


Figure 4: Script identification as part of an intelligent document analyzer

3 Image Quality Metrics and Artifact Detection

In this section we explain the design of our current

script identification engine (in the next two subsections) and additional artifact detection (in the third subsection) corresponding to the OCR Accuracy Predictor block in Figure 4 above. Both methods use feature-based classification, where the classifier is a standard k-nearest-neighbor (K-NN) classifier [17].

3.1 Feature Set for Detecting Merged Characters

Our existing script identification engine has the ability to compute 72 different types of features from a binary image. These features are computed for each individual cluster of adjacent black pixels, or *connected component*. For an ideal document, most such components would be individual character glyphs (exceptions include the letter “i”, which has two components). In a document with significant character-merge artifacts such as shown in Figure 5 below, many of the components will be pairs or groups of adjacent characters. In a document with significant broken-character artifacts such as shown in Figure 7 on the next page, many of the components will be partial character glyphs.

A federal appeals court has ruled that with problems that make it "constituti ruled Thursday that the Child Online I posting information inappropriate for I

Figure 5: Example of degraded text in the “merged characters” category

Given a binary document image, a variety of feature measurements have been proposed for characterizing the 2D shapes present in the image, i.e., the individual components. By far the most popular 2D shape descriptors are the Cartesian moments [12]:

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) dx dy \quad (2)$$

where $f(x,y)$ is the pixel in the character image at position (x,y) . For binary images f is a two-valued function. These moments are sensitive to character placement, orientation, and scale. Typically, nine moments are computed ($p,q=0,1,2$). To compensate for sensitivity to character position, centralized moments are introduced:

$$\mu_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy \quad (3)$$

which gain invariance to character position in the image by subtracting the mean. Again, nine features are computed based on the centralized moments. To remove the sensitivity to scale, normalized moments are computed:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{pq}^\gamma} \quad \text{where } \gamma = \frac{p+q}{2} + 1 \quad (4)$$

which scale each centralized moment by the maximum possible value. Finally, to compensate for changes in rotation, Hu moments [12] are used. Hu moments encode the concept of the shape principal axis to achieve rotational invariance (this is analogous to estimating the principal axis of the shape and performing a rotation to a standard orientation). These moments tend to break down when dealing with rotationally symmetric shapes. Additional moments, such as Zernike and Legendre, have not yet been incorporated into our feature set.

When dealing with moments when either p or q is zero, the measurements are a parametric representation of the marginal distributions. For instance, when p is zero, the centralized moment is proportional to the variance of the pixel values projected onto the vertical axis. Likewise, the Cartesian moment is proportional to the mean. Therefore, these features encode, parametrically, the same information found to be useful for script detection in [8].

While several of the moment features examined are robust under translation, rotation, and scaling, they can be sensitive to degradations in image quality. Such degradations are often present when dealing with poorly scanned documents or faxes. Additional features based on shape can also be used to reduce sensitivity to image noise. An example of such a feature is perimeter, which counts the number of pixels on the boundary of the character. To normalize this value, compactness is used:

$$compactness = \frac{4\pi \cdot area}{perimeter} \quad (5)$$

Other shape features capture curvature, holes, the lengths of the major and minor axis, eccentricity, elongation, and convexity. The final category of features in our set is co-occurrence features [13]. These features are based on measuring how often corresponding positions in the image share the same pixel value. Co-occurrence features are considered textural features and capture notions of symmetry, holes, and repetitive structure. The first step in computing these features involves computing a large matrix representing counts of co-occurring pixel values. To achieve rotation invariance, this matrix is computed at several different orientations. Twenty-seven feature measurements are extracted as a parametric representation of this matrix: mean, variance, entropy, etc.

3.2 Selection of Best Features for Merged Character Detection

The previous section discussed a set of 72 generic features that have been found to be useful in pattern

recognition. To determine which features are most useful for the current task, subset feature selection is used to isolate the most discriminant features.

Feature filters are a technique to isolate good subsets of feature types without using expensive search algorithms. The idea is to compute statistics over the features that indicate which features are better than others. Those features with a low weighting (as set by a threshold) are discarded. Perhaps the most widely used filters are the RELIEF algorithm [14] and its modification for continuous multivariate data, called RELIEF-F [15]. These algorithms update a weighting vector based on the nearest hit (the closest instance in feature space having the same class label) and the average of all of the nearest misses (the closest instance in feature space to each of the other classes). Figure 6 shows the pseudocode for the algorithm.

This algorithm treats the features as dependent, so combinations of features are examined (due to the search for hits and misses being over all features). RELIEF-F produces higher weightings for features that have instances close to neighbors of the same class and farther from instances with different class labels. Likewise, the weights for features that have instances closer in feature space to the wrong class label are decreased. In practice the complexity of the algorithm is $O(N^2)$, rather than $O(2^N)$, since the nearest-neighbor computation will dominate. The approach has been used successfully in many large subset feature selection tasks [16] and shows promise here.

```

Set all weights  $W[A] := 0$ ;
For  $l := 1$  to num_instances do
  Find nearest hit  $H$  and nearest misses  $M$ ;
  For  $A := 1$  to num_attributes do
    Sum := 0;
    For  $C := 1$  to num_classes do
      If  $C \neq \text{ClassOf}(l)$  then
        Sum := Sum + dist( $A, l, M[C]$ );
      EndIf
    EndFor
     $W[A] := W[A] - \text{dist}(A, l, H) + \text{Sum}$ ;
  EndFor
EndFor

```

Figure 6: Pseudocode for RELIEF-F algorithm

3.3 Detection of Broken Characters

While the script identification engine can be successfully retrained to handle detection of merged characters, detecting broken characters, such as shown in Figure 7, requires a different approach. This is due to the fact that our script identifier makes an implicit assumption that most characters in the alphabet consist of just one connected component, and that ignoring dots and diacritics on some characters still makes them recognizable.

A well-regulated militia having become keep and bear arms against one another afforded by the poor or those preferred

Figure 7: Example of degraded text from the “broken characters” characters category

To design a new classifier for detecting broken characters we again turn to script identification as the starting point. This time, however, we seek properties not on the individual component level, but on the word level. The work of Suen [8] serves as the inspiration. The authors of that work pursued the task of separating documents into script and language classes. They converted each word into a sequence of shape codes, based on the position of each constituent connected component’s position with respect to the base line. The result of this transformation was a text document with shape codes substituted for each character, which could be reliably assigned to a language class given an appropriately trained classifier.

One can predict that if a document contains numerous broken characters, the corresponding connected components would contribute shape codes resulting in an unlikely language statistic. However, one can also begin by validating a simpler hypothesis. Specifically, we begin by considering the total number of connected components per word as a random variable and ask whether the probability distribution of this variable can serve as a discriminating feature.

We implemented a word segmentation routine using standard projection approaches [11]. The underlying assumption was that the degradation artifacts (broken or merged characters) did not affect the space between individual words, such that they remained separable.

Figure 8 below shows three distribution curves (normalized as probabilities) computed for sample “Broken”, “Clean”, and “Merged” document images from our corpus.

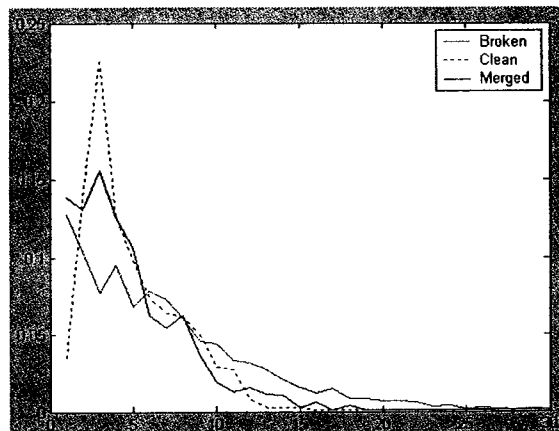


Figure 8: Connected-component per word distribution

The graph shows that most of the mass of the distribution for the clean and merged documents is

contained within words with less than 10 to 12 connected components. The peak is around three connected components, such as for the word “the”. This indicates the metric is not appropriate for separating merged from clean documents. By contrast, the distribution corresponding to the document with numerous broken characters is significantly different. It has a heavy tail and its peak is much lower than the other two distributions.

To implement a classifier (e.g. K-NN) to detect the presence of broken characters we need to design a distance function between training and test samples. We chose the Kullback-Leibler distance [17], also known as the relative entropy. The KL distance between two discrete distributions $p(i)$ and $q(i)$ is defined as

$$d_{KL}(p, q) = \sum_i p(i) \ln \frac{p(i)}{q(i)} \quad (6)$$

The expression is only valid if the two distributions have the same support, so in the implementation we added one to each empty histogram bin. One can prove that if

$$\sum_i p(i) = 1 \text{ and } \sum_i q(i) = 1 \quad (7)$$

then

$$d_{KL}(p, q) \geq 0 \quad (8)$$

with the distance being zero iff the two distributions are identical. (Kullback-Leibler distance is not a distance in the mathematical sense, since it is neither symmetric, nor obeys the triangle inequality.)

4 Experimental Setup

Our 65-document corpus was generated in-house using text collected from articles on news websites as well as e-texts published by the Gutenberg Project. These were formatted as individual pages and printed at 600 dpi on an HP LaserJet 4M Plus. Faxing documents between a Xerox Telecopier 7024 and an HP Officejet d155xi generated the degraded data (merged and broken). Faxing the clean documents from the HP to the Xerox generated broken characters, while faxing the clean originals in the opposite direction generated merged documents. This process produced three sheets of paper (clean, broken, and merged) for each ground truth document.

Each sheet of paper was scanned using a Hewlett-Packard 5960c in grayscale at 300 dpi. For each set of three images scanned (clean, merged, and broken) one was selected at random for the study and binarized using an automatic thresholding method. The resulting images were passed to our two OCR systems, Sakhr’s Automatic Reader 6.0 and Readiris Pro 8.0.

5 Rule-Based OCR Selection

A remaining concern after the automatic labeling of individual page images as “Clean”, “Broken”, or “Merged”, is whether different OCRs in fact handle such documents differently. As a simple test, we ran our document corpus through our two OCR packages and plotted the result in the scatter plot shown in Figure 9 below.

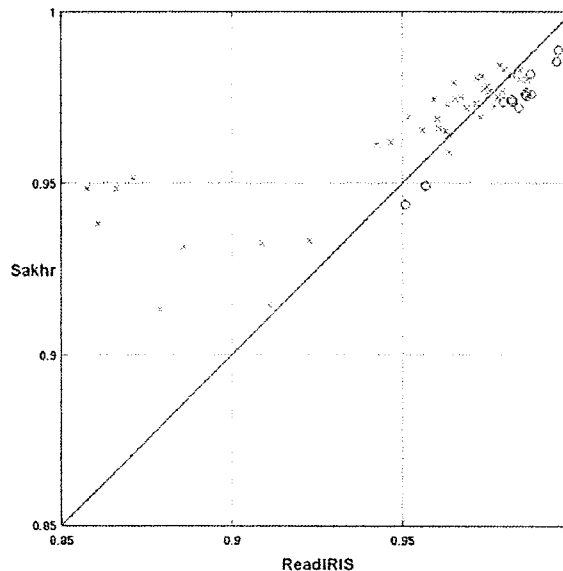


Figure 9: Scatter plot showing Sakhr vs. Readiris accuracy, with circles indicating clean documents and X’s indicating documents with broken or merged character degradation

The scatter plot shows that there is a slight, but significant distinction in the two OCR’s abilities: Sakhr performs better on degraded documents and Readiris is more accurate on clean documents. This result provides the basis for a very simple OCR selection rule for our limited corpus: If page is labeled “Clean” use Readiris, otherwise use Sakhr.

Table 1 shows the average accuracy of each OCR on each of the classes of document quality we studied

Table 1: OCR performance on the document corpus

| | OCR-1 | OCR-2 |
|---------|--------|--------|
| Broken | 0.8916 | 0.9370 |
| Merged | 0.9464 | 0.9666 |
| Clean | 0.9814 | 0.9727 |
| Average | 0.9398 | 0.9588 |

(More descriptive results can be presented using methodology developed in [18]). OCR-2 (Sakhr) performs significantly better on broken characters and moderately better on merged characters. OCR-1 (Readiris) performs slightly better on clean documents. The final row of the table shows the overall average, across all documents in the corpus. Note that while

OCR-2 performs better than OCR-1 on average, clean documents make up only 22.4% of the corpus.

6 Results of Automatic Page Quality Assessment

6.1 Artifact Detector 1: Broken versus Merged/Clean

The artifact detector described in section 3.3 was trained on several pages produced in a similar manner to how the testing data was produced. The detector assigned one of two labels to a given document: "Broken" or "Merged/Clean". The broken documents fed directly into the best OCR for broken data (OCR-2). The merged/clean data is fed to the second artifact detector (the script identification engine). Table 2 shows the confusion matrix for the 65 page test document corpus.

Table 2: Classification results on the broken versus merged/clean problem computed over individual connected components for all documents

| | Broken | Merged/Clean |
|--------------|--------|--------------|
| Broken | 0.77 | 0.23 |
| Merged/Clean | 0.03 | 0.97 |

This artifact detector performed very well in detecting merged/clean pages (97%) and reasonably well in detecting broken pages (77%). Since the 23% of broken pages which were mislabeled merged/clean are passed to another classifier, some of those do in fact end up being routed to the correct OCR.

6.2 Artifact Detector 2: Merged versus Clean

The RELIEF-F algorithm was applied to a training set consisting of merged and clean data. The goal was to determine which of the 72 feature measures are most appropriate for this task. While all of the features implemented have shown promise in other domains, some clearly provide better discrimination in this

Table 3. Five best features from RELIEF-F

| Feature | Category | RELIEF-F Score |
|----------------------------|---------------|----------------|
| Measure of Correlation (X) | Co-occurrence | 543 |
| Curl | Shape | 498 |
| Fiber Length | Shape | 463 |
| Measure of Correlation (Y) | Co-occurrence | 444 |
| Variance | Co-occurrence | 340 |

context. We retain the top 5 features based on their rank. Table 3 shows the highest ranked features.

The decision to choose only five features for classification was based on a visual analysis of the data. Scatter plots were generated for pairs of features ranked high by RELIEF-F. Figure 10 shows an example of a scatter plot for the two best-ranked features. In this example, there is an obvious separation between the clear class (red) and merged class (green). Beyond five features the separation was not as noticeable.

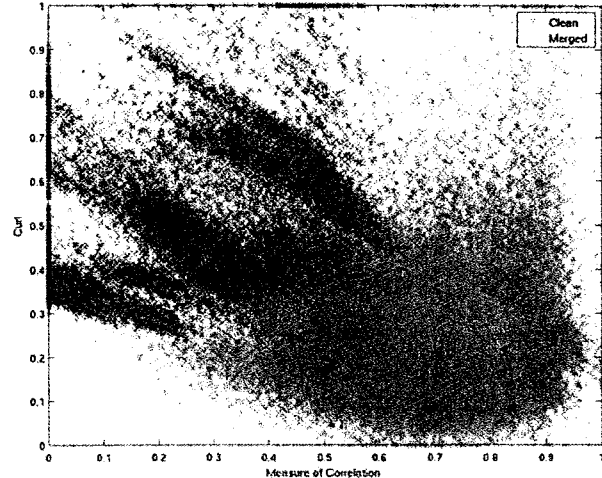


Figure 10: Top two RELIEF-F features for the Merged/Clean problem

These five features were used to train a K-NN classifier. The training set was constructed by printing a typed page containing 20 instances of each letter of the alphabet (in both upper and lower case). The page was then scanned to produce the clean training set, and subjected to the same process that produced the merged testing dataset. The connected components algorithm was then run to generate glyphs. In total, the training set contains on the order of 5000 patterns.

From each pattern in the training set, the five features listed in Table 3 were extracted and stored. The testing set (which does not overlap the training set) was then used to evaluate the algorithm (connected components, then feature extraction, then K-NN classification). There were over 100,000 testing patterns (glyphs extracted by connected components) in the testing set.

Table 4 shows the accuracy of the K-NN classifier. The average performance is quite good at the glyph level. This shows the ability to detect document corruption at a very fine scale (much smaller than an entire page).

Table 4: Classification results on the merged/clean problem computed over individual connected components for all documents

| | Clean | Merged |
|--------|-------|--------|
| Clean | 0.94 | 0.06 |
| Merged | 0.04 | 0.96 |

Given the expected level of correct classification at

the connected components level, we can estimate the number of connected components that must be examined to correctly label the entire page. To develop such a model we will model the assignment of the correct label to an individual connected component as a binomial process. The intuition is that at the component level the classifier will either succeed or fail to give the correct label. If the probability of successful classification p can be estimated, then the Binomial probability density function (pdf) allows an estimate of the chance of seeing k successes in n glyphs:

$$P(k) = \binom{n}{k} p^k (1-p)^{n-k} \quad (9)$$

Since we use voting over a series of glyphs to assign labels to pages, the probability of seeing more correct labels than incorrect labels ($k > n/2$) can be computed based on the cumulative density function of the Binomial equation:

$$P(k \geq n/2) = \sum_{k=0}^{n/2} \binom{n}{k} p^k (1-p)^{n-k} \quad (10)$$

A nice property of viewing the problem in this manner is that the total number of connected components that need to be examined can be estimated given knowledge of p .

Table 5 shows the relationship of p and n for $P(k \geq n/2)$ where n is the number of components, and p is the likelihood of assigning the correct label.

Table 5: Estimating the number of connected components that must be examined to determine the correct document label

| n/p | 10 | 20 | 30 | 40 | 50 | 60 | 70 |
|------|------|------|------|------|------|------|------|
| 0.01 | 0.09 | 0.18 | 0.26 | 0.33 | 0.39 | 0.45 | 0.51 |
| 0.1 | 0.65 | 0.87 | 0.95 | 0.98 | 0.99 | ~1.0 | ~1.0 |
| 0.2 | 0.89 | 0.98 | 0.99 | ~1.0 | ~1.0 | ~1.0 | ~1.0 |
| 0.4 | 0.99 | 0.99 | ~1.0 | ~1.0 | ~1.0 | ~1.0 | ~1.0 |
| >0.5 | ~1.0 | ~1.0 | ~1.0 | ~1.0 | ~1.0 | ~1.0 | ~1.0 |

Assuming that each component is independent, even with a low probability of correct label, voting over more components will reduce the error across the entire page.

6.3 Performance Bounds Modeling

The multi-stage artifact detector can be used to estimate the total performance accuracy. Figure 11 shows the possible paths through the system. The percentages correct for each OCR are obtained from

Table 1. The weights on the arrows are based on the two artifact detector classification rates. For instance, when considering documents containing broken text,

the current detector will choose the best OCR algorithm for the task 77% of the time and use the worst OCR 23% of the time. This is actually an estimate on the lower bound since the true estimate of the performance of the worst OCR on broken documents is really a function of whether it was subsequently assigned the label of merged or clean.

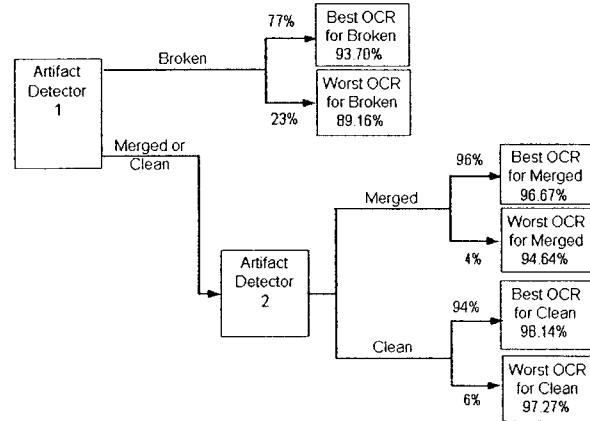


Figure 11: Estimating the overall system performance

We can compare the performance model to an algorithm that randomly selects between OCRs (i.e., uses no detection and therefore is equally likely to choose either OCR). Table 6 shows that biasing the choice of classifier using the artifact detector improves aggregate accuracy. The expected performance of the our approach is simply the weighted performance in using the wrong OCR (an incorrect decision was made):

$W = P * Max(OCR1, OCR2) + (1 - P) * Min(OCR1, OCR2)$ where W is the performance measure and P is the likelihood the correct decision was made. In the case of a page labeled broken, we have:

$$Broken = 0.77 * 0.937 + 0.23 * 0.8916 = 0.926$$

Table 6: Comparing random decisions to our detection bias mechanism

| | Random | Using Detectors |
|---------|----------|-----------------|
| Broken | 0.914302 | 0.926586 |
| Merged | 0.956488 | 0.965806 |
| Clean | 0.977076 | 0.980918 |
| Average | 0.949289 | 0.95777 |

7 Conclusions

We have tackled the problem of OCR accuracy prediction and selection using image features developed for script identification. Although our intent was to find a learning scheme for an existing script identification engine, not all artifacts could be detected that way,

leading to a custom approach to measure the presence of broken characters. It is still possible that the two schemes can be "rolled in" into a single module capable of serving two purposes at once, but this is just a conjecture.

Our system was validated on a relatively small document corpus. However, applying two OCR engines from two different vendors to this corpus showed that even the latest commercial systems do differ in their ability to handle very basic degradation artifacts (light and heavy print), underscoring the need to detect such artifacts in the context of the document enhancement pipeline.

Acknowledgements

This work was supported by the Army Research Labs, Adelphi, MD, under contract DAAD17-02-C-0043.

References

- [1] L.R. Blando, J. Kanai, and T.A. Nartker, "Prediction of OCR Accuracy using Simple Image Features," *ICDAR'95*.
- [2] J. Gonzalez, J. Kanai, and T.A. Nartker, "Prediction of OCR Accuracy using a Neural Network," *Document Analysis Systems II*, World Scientific Publishing, 1997.
- [3] M. Cannon, J. Hochberg, and P. Kelly, "QUARC: A Remarkably Effective Method for Increasing the OCR Accuracy of Degraded Typewritten Documents," *SDIUT'99*.
- [4] P. Thouin, Y. Du, and C-I. Chang, "Low Resolution Expansion of Gray Scale Text Images Using Gibbs-Markov Random Field Model," *SDIUT'01*.
- [5] J. F. McNamara, D. W. Casey, R. W. Smith, and D. S. Bradburn, "A Classifier for Evaluating the Effects of Image Processing on Character Recognition," *Image Algebra and Morphological Image Processing (IV)*, 1993.
- [6] J. Hochberg, P. Kelly, T. Thomas, and L. Kerns, "Automatic Script Identification from Document Images Using Cluster-Based Templates," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 176-181, Feb. 1997.
- [7] T. N. Tan, "Rotation Invariant Texture Features and Their Use in Automatic Script Identification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 7, pp. 751-756, July 1998.
- [8] C. Y. Suen, S. Bergler, N. Nobile, B. Waked, C. P. Nadal, and A. Bloch, "Categorizing Document Images into Script and Language Classes," *International Conference on Advances in Pattern Recognition, ICAPR'98*, Plymouth, UK, Nov. 1998.
- [9] D. S. Lee, C. R. Nohl, and H. S. Baird, "Language Identification in Complex, Unoriented, and Degraded Document Images," *IAPR Workshop on Document Analysis Systems*, pp. 76-98, Malvern, Pennsylvania, Oct. 1996.
- [10] J. Ding, L. Lam, and C. Suen, "Classification and Identification of Multi-lingual Documents," *International Workshop on Performance Evaluation Issues in Multi-lingual OCR*, (in conjunction with ICDAR'99), Bangalore, India, Sept. 19, 1999.
- [11] J. Ha, I. T. Phillips, and R. M. Haralick, "Document Page Decomposition by the Bounding-Box Projection Technique," *ICDAR'95*.
- [12] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, 1992.
- [13] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Texture Features for Image Classification," *IEEE Trans. Systems, Man, and Cybernetics*, SMC-3, 1973.
- [14] K. Kira and L. Rendell, "The Feature Selection Problem: Traditional Methods and a New Algorithm," *10th National Conference on Artificial Intelligence (AAAI'92)*, 1992.
- [15] I. Kononenko, "Estimating Attributes: Analysis and Extensions of RELIEF," *European Conference on Machine Learning*, 1994.
- [16] J. Bins and B. Draper, "Feature Selection from Huge Feature Sets," *International Conference on Computer Vision*, 2000.
- [17] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed., John Wiley & Sons, 2000.
- [18] T. Kanungo, G.A. Marton, and O. Bulbul, "Paired Model Evaluation of OCR Algorithms," *SPIE Conference on Document Recognition and Retrieval (VI)*, San Jose, CA, 1999.

OCR and OCR Corrections

A Survey of Retrieval Strategies for OCR Text Collections

Steven M. Beitzel, Eric C. Jensen, David A. Grossman
Information Retrieval Laboratory
Department of Computer Science
Illinois Institute of Technology
{steve,ej,grossman}@ir.iit.edu

Abstract

The importance of effectively retrieving OCR text has grown significantly in recent years. We provide a brief overview of work done to improve the effectiveness of retrieval of OCR text.

Introduction

As electronic media becomes more and more prevalent, the need for transferring older documents to the electronic domain grows. Optical Character Recognition (OCR) works by scanning source documents and performing character analysis on the resulting images, giving a translation to ASCII text, which can then be stored and manipulated electronically like any standard electronic document. Unfortunately, the character recognition process is not perfect, and errors often occur. These errors have an adverse effect on the effectiveness of information retrieval algorithms that are based on exact matches of query terms and document terms. Searching OCR data is essentially a search of “noisy” or error-filled text.

We briefly survey approaches to searching OCR text. These include: defining an IR model for OCR text collections, using an OCR-error aware processing component as part of a text categorization system, auto-correction of errors introduced by the OCR process, improving string matching for noisy data, and issues in cross-language retrieval of image data. We also note that an excellent survey of the more general area of indexing and retrieving document images can be found in [Doer98]. Other recent papers surveying current approaches in OCR Information Retrieval appeared in the 2002 SIGIR workshop on OCR Information Retrieval¹. The following sections of this paper will examine each of these approaches, and give a summary of the progress made in each area.

IR models for OCR text

Most work in the field of OCR Information Retrieval has been relatively recent. A primary reason for this is that, especially in early years, obtaining a sufficient quantity of data on which to test was very difficult. To get around this problem, Croft and colleagues published one of the first studies of the effects of OCR data on IR by using simulated OCR collections [Croft94]. This study found that for high quality OCR conversion, not much degradation in retrieval effectiveness was encountered, however the retrieval of

¹ <http://www.info.uta.fi/sigir2002/html/ws2.htm>

short and low-quality documents was adversely affected. Lopresti and Zhou examined the performance of several models of IR on simulated OCR data, and were able to show that it was plausible to use modified approaches (they made use of fuzzy logic and approximate string matching) to increase effectiveness on noisy data [Lopr96]. These conclusions led to the development of models of IR designed specifically for operating on a body of OCR text. Some of the initial work on developing a model of Information Retrieval specifically suited for operating on a collection of OCR text was done by [Mitt95]. Their study involved the development of a term weighting scheme for the probabilistic model of information retrieval. The motivations for this work arose from conclusions reached in [Crof94, Tagh94a, Tagh94b, Tagh96] (and later explored in depth by Mittendorf and colleagues in [Mitt96]), which show that, although retrieval performance is not generally adversely affected by errors in an OCR text collection, performance degradation is often observed in cases where there are few documents in the collection, or the documents in question are very short. It was also observed that the presence of errors in OCR text tends to lead to unstable and unpredictable retrieval performance. In an effort to develop a retrieval model that circumvents these limitations, they incorporate the occurrence probabilities of several kinds of typical OCR errors into their term-weighting schemes. In a set of experiments on a collection of very short documents, they achieved a 23-30% improvement in retrieval effectiveness by using a form of the probabilistic model. Their general equation for the Retrieval Status Value (RSV) of a document is given in Equation 1 below:

$$RSV(q, d_j) \equiv \sum ff(\varepsilon_i, q)ff(\varepsilon_i, d_j)/\lambda_j$$

Where:

| | |
|--------------------------|--|
| q | = query |
| d_j | = document |
| $ff(\varepsilon_i, d_j)$ | = feature frequency in document |
| $ff(\varepsilon_i, q)$ | = feature frequency in query |
| λ_j | = number of occurrences of feature frequency in document |

Equation 1 – Retrieval Status Value for OCR-IR

The referenced work presents a number of methods for estimating various feature frequencies and the probabilities of their occurrence.

Further work has been done in adapting the probabilistic model of IR to handle error-ridden OCR text. Ohta enhanced the probabilistic model to take advantage of expected errors in OCR text [Ohta97]. Specific character transformations and character occurrence bigrams were used to generate candidate terms for each “true” search term. Documents retrieved by each candidate term are then evaluated for inclusion into the final result set. This resulted in minor improvements in recall for moderate quality OCR documents. Another study was performed in [Hard97]. This study makes use of n-grams to overcome the problems introduced by OCR text, which is a quite fitting solution, when considering that a large percentage of typical OCR errors involve mis-identifying a sequence of one or two characters within a given word. Their basic approach defines a

set of “binding operators” over the constituent n-grams of a word. The goal of these operators is to be strict enough to exclude noise and non-relevant information from the top documents, while lenient enough to prevent elimination of relevant data when a particular constituent n-gram is missing. The experimentation performed in the study was designed to discover which operators would maximize retrieval performance, and it was found that the “passage5” operator, which ranks documents containing n-gram components within windows of five word positions, while allowing the windows to cross word boundaries was the most effective approach. It is theorized that this is to be expected because of the extremely common OCR error wherein spaces are added to the text in improper locations. An example of this can be seen with the word “environmental”, which is broken up into the ngram components (en env envi ironm onm ment tal al). The length of this word makes it a very likely candidate for incorrect identification during the OCR process via the insertion of one, or several spaces, however, the passage5 operator binds these components together loosely over a large range that may cross word boundaries, helping to mitigate this problem. In their experiments, operating on four databases that were randomly degraded using data developed at UNLV [Rice93], an improvement ranging from 5-38% was observed, depending on the test collection. The authors used a similar approach to aid in query term expansion. They used n-grams to discover candidate expansion terms that were a match or a near-match to terms in the original query. As expected, a further improvement of 9-18% was observed when using n-grams for query expansion. Very recently, work has been done that takes common OCR errors into account when generating a language model [Jin02]. This language model can then be used to approximate an “uncorrupted” version of a particular document, and it can be used for retrieval in a language modeling approach. The authors found significant improvement when using this approach over other approaches that explicitly correct each error found in the source documents.

Processing OCR Text for Categorization

In addition to document retrieval, there are other areas of IR that must effectively handle OCR text. One such area is Text Categorization, wherein a group of documents are examined and assigned to a set of categories, typically to aid in document browsing or visualization. Some of the early work in this area is presented in [Hoch94]. This study does not deal directly with mitigating problems introduced by OCR Text, rather, they describe an approach to the development of an automatic indexing and classification system that uses advanced morphological analysis of the text, along with term frequency analysis, index term weighting, and training of the classifier based on a previously-defined document model [Deng92]. The results of this study indicate that classification performance was inhibited by the degradation of index terms from the OCR process. No solution to the problem is given, although it is reasonable to assume that the incorporation of a specialized term weighting scheme for OCR documents, such as the ones described above, would help to improve performance. Evidence of this assumption is presented in [Cavn94] and [Junk97], wherein advanced techniques such as n-gram processing and morphological analysis are used to aid in reducing the effect of imperfections introduced by the OCR process on retrieval effectiveness.

A survey of common techniques used to enhance effectiveness in text categorization can be found in [Seba02]

Auto-Correction of OCR errors

One valuable area of research involves post-processing systems that work to correct the errors that are introduced in the OCR process. This has far-reaching implications, as being able to efficiently compensate for OCR errors allows conventional IR techniques to be used without experiencing degradation in effectiveness. Some early work in this area is discussed in [Liu91]. This study examines and classifies each type of error that can be introduced by the OCR process. Furthermore, it identifies which errors are the most typical and most likely to introduce confusion in the resulting documents. Several techniques for correcting the errors are discussed as well. Dictionary lookups on candidate terms are one of the most course-grained techniques used, and they help to identify words that have likely been corrupted. In addition, terms in the source text are broken up into digrams, and a frequency matrix is kept to help identify which character sequences are indicative of errors. Based on this, adaptive character conversion maps are constructed that allow the system to identify a character sequence likely to be in error, and automatically correct it by performing a lookup in the map and replacing it with the corrected version. These automated techniques were performed in concert with user interaction, and resulted in a significantly improved final text that did not require nearly as much user intervention as prior corrective approaches.

Another study involving the use of a post processing system for OCR error correction was performed in [Tagh94a, Tagh94b]. They used techniques similar to the ones mentioned above, with the addition of a clustering technique that grouped collections of misspellings in with their correctly spelled target term. After frequency analysis on the term spelling clusters eliminated unlikely candidates, each misspelled term was replaced with its correctly spelled counterpart. A more complete overview of error correction techniques in post processing systems can be found in [Kuki92].

Improved String Matching on Noisy Data

For applications where it is desirable to find all occurrences of a particular term, there is the notion of exact string matching. When the data is noisy or corrupted, as is the case with OCR text, exact string matching becomes difficult. This problem has been approached by training language models to recognize terms that are improperly spelled, as done in [Coll01]. A more detailed survey of string matching approaches can be found in [Nava01].

OCR Issues in Cross-Language Retrieval

Performing Information Retrieval on OCR documents in non-English languages provides some interesting and unique challenges. Oard and colleagues have implemented a full system for cross-language retrieval, and have extended that system to support text from document images [Oard99]. Basically, this system proposes the use of character-confusion statistics and character-class recognition algorithms that are specific to the target language in order to mitigate the ambiguities and errors introduced into a text by the OCR process. More recently, Darwish and Oard have focused on retrieval of OCR'd

Arabic text, and have found that using a combination of light-stemming approaches and character n-grams for the selection of candidate index terms generally provides the most improvement in retrieval effectiveness, and is robust over a variety of OCR errors [Darw01, Darw02].

Summary

We have briefly surveyed current techniques in use to facilitate information retrieval on collections of OCR text. There are a large number of proposed techniques and models available for use, and hopefully in the future a generalized solution that takes on aspects of all available techniques will be available to members of the Information Retrieval community.

References

- [Cavn94] W. Cavnar and J. Trenkle. N-Gram based text categorization. In Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval, pages 161-175, Las Vegas, NV, 1994.
- [Crof94] W. B. Croft, S. M. Harding, K. Taghva and J. Borsack. An Evaluation of Information Retrieval Accuracy with Simulated OCR Output. Proceedings of the Symposium on Document Analysis and Information Retrieval, 1994.
- [Coll01] Kevyn Collins-Thompson and Charles Schweizer and Susan Dumais. Improved string matching under noisy channel conditions. Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM), 2001.
- [Darw01] Darwish, Kareem, D. Doermann, R. Jones, D. Oard, and M. Rautiainen. TREC-10 Experiments at Maryland: CLIR and Video. TREC-2001, 2001.
- [Darw02] Kareem Darwish and Douglas W. Oard. Term Selection for Searching Printed Arabic. Proceedings of the Twenty-Fifth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 2002.
- [Deng92] A. Dengel, R. Bleisinger, R. Hoch, F. Fein, F. Hones. From Paper to Office Document Standard Representation. IEEE Computer, vol. 25, no. 7, 1992, pp. 63-67.
- [Doer98] David Doermann. The Indexing and Retrieval of Document Images: A Survey, The Journal of Computer Vision and Image Understanding: CVIU, Volume 70, Number 3, 1998.
- [Fras01] Paolo Frasconi and Giovanni Soda and Alessandro Vullo. Categorization for multi-page documents: A Hybrid Naive Bayes HMM Approach. Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries, 2001.

[Hard97] Stephen M. Harding and W. Bruce Croft and C. Weir. Probabilistic Retrieval of {OCR} Degraded Text Using N-Grams. Proceedings of the European Conference on Digital Libraries (ECDL), 1997.

[Hoch94] Rainer Hoch. Using IR techniques for text classification in document analysis. Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, 1994.

[Junk97] M. Junker and R. Hoch. Evaluating OCR and non-OCR text representations for learning document classifiers. Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), 1997.

[Kuki92] Karen Kukich. Technique for automatically correcting words in text, ACM Computing Surveys, Vol 24, No. 4, 1992.

[Liu91] Lon-Mu Liu and Yair M. Babad and Wei Sun and Ki-Kan Chan. Adaptive post-processing of OCR text via knowledge acquisition. Proceedings of the 19th Annual Conference on Computer Science, 1991.

[Lopr96] D. Lopresti and J. Zhou. Retrieval Strategies for Noisy Text. Proceedings of the Symposium on Document Analysis and Information Retrieval, 1996.

[Mitt95] Elke Mittendorf and Peter Schäuble and Páraic Sheridan. Applying probabilistic term weighting to OCR text in the case of a large alphabetic library catalogue, Proceedings of the 18th Annual international ACM SIGIR Conference on Research and Development in Information Retrieval, 1995.

[Mitt96] E. Mittendorf and P. Schauble. Measuring the effects of data corruption on information retrieval. Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR), 1996.

[Nava01] Gonzalo Navarro. A guided tour to approximate string matching, ACM Computing Surveys, vol 33, no. 1, 2001.

[Oard99] Douglas W. Oard. Issues in Cross-Language Retrieval from Document Image Collections. In Proceedings of the 1999 Symposium on Document Image and Understanding Technology, 1999.

[Ohta97] M. Ohta, A. Takasu, and J. Adachi. Retrieval Methods for English text with misrecognized OCR characters. Proceedings of the International Conference on Document Analysis and Recognition, 1997.

[Rice93] S. Rice, J. Kanai, and T. Nartker. An Evaluation of Information Retrieval Accuracy. In UNLV Information Science Research Institute Annual Report (1993), 9-20.

[Seba02] Fabrizio Sebastiani. Machine learning in automated text categorization, ACM Computing Surveys (CSUR), 34:1, 2002.

[Tagh94a] Kazem Taghva and Julie Borsack and Allen Condit. Results of applying probabilistic IR to OCR text, proceedings of the seventeenth annual international ACM-SIGIR conference on Research and development in information retrieval, 1994.

[Tagh94b] Kazem Taghva, Julie Borsack and Allen Condit. An Expert System for Automatically Correcting OCR Output, in Proceedings of the SPIE – Document Recognition, 1994.

[Tagh96] Kazem Taghva and Julie Borsack and Allen Condit. Evaluation of model-based retrieval effectiveness with OCR text. ACM Transactions on Information Systems (TOIS), 14:1, 1996.

OCR Accuracy and Retrievability of Post-processed Documents

Tom Nartker, Kazem Taghva, and Julie Borsack

Information Science Research Institute
University of Nevada, Las Vegas
tom@isri.unlv.edu

Abstract

This paper describes two tests that measure the quality of a document conversion system applied by the Department of Energy (DOE) for the Licensing Support Network (LSN). The tests measure OCR accuracy using a new metric, non-stopword accuracy. We then show that retrievability for the DOE system is comparable to a 99.8% manually corrected version of the same collection.

1 Introduction

The Information Science Research Institute (ISRI) at the University of Nevada, Las Vegas (UNLV) was tasked with evaluating the performance of the Department of Energy (DOE) document conversion system used to prepare documents for the Licensing Support Network (LSN). Their current system applies automatic zoning and OCR followed by automated post-processing using a system developed at ISRI called, *MANICURE*. These documents will be subsequently loaded into an information retrieval (IR) system for document discovery during the licensing proceedings. This paper reports on the studies performed and gives a summary of the results of two performance criteria applied in our evaluation: *non-stopword accuracy* and *retrievability*.

There are two approaches to evaluating the performance of document conversion systems. One approach is to measure the accuracy of the textual output using the metric, average character accuracy[2][1]. A second approach is to measure the performance of the system that will make use of the output text. As mentioned, the textual output from the OCR process will be used to build the index for an IR system that will aid in the task of document discovery. The appropriate performance measure for IR systems is retrievability (i.e., precision and recall)[5]. Thus, to provide a thorough evaluation of complete system performance, two different studies (a character accuracy study and a retrievability study) were conducted[4][13].

The first series of tests [4] were aimed at redefining the notion of character accuracy. Raw character accuracy measures the accuracy of the OCR engine for the set of all characters produced. For retrieval purposes though, the more appropriate measure is the character accuracy of *non-stopwords* which we apply in this performance evaluation task. The second series of tests [13] were designed to address retrievability. We designed an experiment to compare documents that had been OCR'd and automatically-zoned to the identical document set that had been manually-zoned and manually corrected to 99.8% character accuracy. These experiments show that the use of automatic-zoning followed by *MANICURE* not only corrects non-stopwords (the words used for retrieval) but also gives retrieval results equivalent to what one could expect from a 99.8% correct collection.

2 About MANICURE

Both tests mentioned in Section 1 apply a post-processing system designed and implemented at ISRI. The heuristics that make up the system were developed through careful analysis of the nature of OCR errors, complications of OCR in IR applications, and the ability to apply document and collection level information to post-processing correction. *MANICURE* (Markup ANd Image-based Correction Using Rapid Editing) then was built to facilitate the task of constructing electronic documents that would be used in other applications like IR[14]. It was designed to take advantage of document characteristics such as word forms, geometric information, and font and spacing between textual objects to mark the logical structure of a document. In addition, *MANICURE* automatically detects and corrects OCR spelling errors using dictionaries, approximation matching, the knowledge of typical OCR errors, and frequency and distribution of words and phrases in a document. We found that *MANICURE*, in its automatic mode, produces electronic docu-

ments which are good for most text analysis and retrieval applications.

Studies on effects of OCR errors on retrieval[12, 9, 10] pointed out that certain advanced functionalities of information retrieval systems, such as ranking could be impacted by some types of OCR errors. In fact, the more advanced retrieval techniques and applications may actually require a higher quality OCR output, particularly less "graphic text" to function optimally. *Graphic text* can be defined as strings of spurious characters that are generated when an OCR device tries to recognize non-textual material such as figures, tables, and equations. MANICURE can be used in both automatic and semi-automatic modes to produce OCR text of higher quality. MANICURE is composed of four modules and an OCR front end. The four modules consist of the parser (*doc_parse*), the logical document tagger (*autotag*), the post processing system (*ppsys*), and a semi-automatic user interface (*rummage*). In this paper, we focus on *ppsys* and its ability to post-process OCR text in preparation for use with an IR system.

The post processing component was devised to detect and correct OCR errors through approximation matching, by consulting the OCR's most common confusions, and by extracting knowledge from the complete document[8]. Since the complete document has already been processed, information about its content can be used to correct it; an OCR device does not have access to all this information when it processes single document pages. For example, a word on the first page may have been misrecognized but another occurrence of the same word on the next page may have been correctly recognized. By having the entire document available, we have more information for correcting misrecognized occurrences.

Ppsys builds an inverted file from *autotag*'s output consisting of a document's words, their frequencies, and their proximities to other words in the document. Dictionaries[15] are used to mark each word as correctly recognized or misspelled. The system then generates statistical phrases to correct misspellings. This correction routine is followed by approximation matching using word frequency information and OCR error information (e.g. *rn* for *m*).

In addition, two sub-modules of *ppsys* assist in OCR error correction: *rmgarbage*[16] and the Acronym Finder Program or *AFP*[15]. *Rmgarbage* aids OCR correction for retrieval by eliminating the graphic text that can cause inflated IR indexes, increased document length, unreliable term frequencies, and ranking variability. Systems that rely on statistical data from a collection for retrieval, categorization, ranking, or other similar document manipulation tasks can be affected by an increase in garbage strings[9]. *AFP* marks special words typi-

cally not found in dictionaries but are known to be valid strings. We observed that OCR devices generally have lower accuracy rates for acronyms[11]. This is due to the fact that these devices rely heavily on the use of lexicons and statistical distribution of character n-grams. Without a method of identifying these special words, *ppsys* would invariably tag them as misspellings and worse, correct them mistakenly. *AFP* locates and defines acronyms in OCR text making these words available to *ppsys* for correction and later for retrieval.

3 OCR Word Accuracy Tests

The task of document preparation for the LSN begins with *character recognition*. The DOE has selected and installed the "Developers Kit 2000" (SDK2000) distributed by the Scansoft Corporation[6]. SDK2000 is based on combined technologies developed by the Calera, Caere, and Recognita Corporations, and is the most popular page reading engine for general purpose use. Because the ultimate purpose for the text produced by the OCR engine was to enter it into an IR system, DOE has applied MANICURE as part of its document conversion process. As previously stated, MANICURE was designed to accept character output from an OCR engine and perform operations on the text that improves retrievability. Thus, the DOE document conversion system is a combination of the SDK2000 OCR engine and the MANICURE post-processing system. For this reason, both the accuracy of OCR output text and of MANICURE output text are reported.

3.1 Measuring Character Accuracy of Textual Output

Character accuracy measures the correctness of every ASCII character on each page and is defined in Equation 1 as:

$$\frac{\text{total_characters} - \text{character_errors}}{\text{total_characters}} \quad (1)$$

Character errors are the sum of character insertions, deletions, and substitutions that are necessary to convert an output character string into the exact *ground-truth string*[1]. The ground-truth string can be defined as the actual string as it appears in the original hard copy document.

Obtaining and comparing OCR generated text to its corresponding ground-truth is more difficult than it may first appear[3]. The task of measuring the accuracy of textual output is complicated by several factors. First, in order to measure the accuracy of a text stream, it is necessary to have a "correct" text stream for comparison. In most cases, the cost of producing the correct or ground-truth character

stream is very high. Not only must each character of the document be manually retyped but each character must be checked and rechecked for correctness. Second, it is necessary to conduct such tests with large numbers of pages. No test of 5, or even 10 pages can be expected to produce statistically significant results. In general, it is preferable to have hundreds of test pages in order to insure significance of the measured results. Third, it is not a trivial issue to determine exactly which accuracy measure is most appropriate for a particular application.

Character recognition is typically measured by standard character accuracy even though many characters in the text have no role in retrievability. For example, punctuation marks, end-of-line hyphenation, and characters in *stopwords* are ignored by an information retrieval (IR) system. So it is important to determine what accuracy metric is the most appropriate measure. We believe that for the DOE document conversion system, the most appropriate metric should be an indicator of retrieval effectiveness. Although the character accuracy of output text is related to retrievability, the conventional definition of character accuracy is not the best measure. Since the conversion output will be used to build the index in an IR system, it is the accuracy of the words that will be indexed that better reflects retrievability. "Word" accuracy is defined in Equation 2 as:

$$\frac{\text{total_words} - \text{word_errors}}{\text{total_words}} \quad (2)$$

In fact, since IR systems normally ignore some specific strings, such as stopwords and numerics, "non-stopword" accuracy is yet a better measure of retrievability. Equation 2 can also be used to calculate non-stopword accuracy by substituting non-stopwords for words.

Since print noise (or any stray marks), numbers, and punctuation marks are NOT indexed by IR systems, they do not affect retrievability. "Conventional" character accuracy though can be profoundly affected by these kinds of characters so it is clear that "non-stopword" accuracy is a much better measure of retrievability. One possible alternative is to measure just the accuracy of the characters used to make up non-stopwords as an alternative to the "conventional" definition of character accuracy. We refer to this measure as the "character accuracy of non-stopwords" and use Equation 1 by replacing "characters" with "characters in non-stopwords." To ensure that the full benefit of using MANICURE is measured, complete documents were used in these tests.

Table 1: Number of Non-stopwords and Characters in the 17 Document Sample

| Document ID | # of Non-stopwords | # of Chars in Non-stopwords |
|-----------------|--------------------|-----------------------------|
| mol199907200407 | 6974 | 56611 |
| mol199911010207 | 9641 | 80684 |
| mol200002170216 | 9595 | 79777 |
| mol200002280529 | 5572 | 47005 |
| mol200004130692 | 4115 | 33855 |
| mol200004140874 | 12379 | 101435 |
| mol200005230155 | 4381 | 37387 |
| mol200005250378 | 13920 | 113193 |
| mol200005260336 | 6318 | 51210 |
| mol200006090266 | 5112 | 44172 |
| mol200006270254 | 7792 | 62586 |
| mol200007250453 | 36713 | 302763 |
| mol200011220005 | 8440 | 70020 |
| mol200012080086 | 4523 | 37402 |
| mol200101250233 | 14247 | 120653 |
| mol200103160002 | 8441 | 69501 |
| mol200104160088 | 6320 | 52870 |
| Average | 9675.47 | 80066.12 |

3.2 Non-stopword Accuracy of DOE's Document Conversion System

In Section 3.1 we pointed out some of the difficulties one may encounter when preparing ground-truth text. The cost of producing accurate ground-truth for even a few typical length DOE documents is extremely high. Thus, finding a low cost method of producing ground-truth was a dominant part of conducting our document level tests. To solve this problem, we selected 17 documents at random from the DOE collection that had Microsoft Word native files. The document ID, the total number of non-stopwords, and the number of characters in these non-stopwords are shown in Table 1.

We developed a process to capture the correct output text directly from Microsoft Word. We also parsed this text to remove all punctuation, most of the digits, and all stopwords¹. A concerted effort was made to retain document identifiers and other "project words" containing digits². Thus, the remaining text contained only English non-stopwords (and project related non-stopwords that might not be in a normal dictionary) forming the basis for com-

¹The stopwords removed were the *Brown Corpus* list of 450 stopwords.

²The criterion for "project words" that were retained was the same as that used by the MANICURE system. Equations, tables, graphs, and other non-textual material were manually removed.

Table 2: Average Character Accuracy for 17 Document Sample

| Generation | RAW OCR | MANICURE |
|------------|---------|----------|
| Orig. | 99.37 | 99.50 |
| Gen 0 | 99.40 | 99.47 |
| Gen 1 | 99.20 | 99.30 |
| Gen 2 | 98.67 | 98.83 |
| Gen 3 | 97.89 | 98.16 |
| Gen 4 | 97.79 | 98.06 |

puting accuracies.

Because the images extracted from native Microsoft Word documents were never printed or scanned, they were completely free of defects associated with either the printing or scanning process. OCR output from these images, however, would not produce results that were typical of the current DOE document conversion. Even if each document were printed and scanned, the images produced would still be of higher quality than the average image from the DOE collection.

We therefore chose not only to print and scan each page of these documents but to produce several generations of photocopies; we chose to measure non-stopword accuracies not only from the original images but also from the first printed and scanned image and from the first, second, third, and fourth generation photocopies of these images. The first image was the "original" image extracted from Microsoft Word. The second image was the first-printed and scanned image and we refer to this generation as "generation 0." The third through sixth images are the first through fourth generation photocopies of these documents. Our best judgement of the average quality of images in the DOE collection is somewhere between a first and second-generation photocopy.

3.3 Non-stopword Accuracy Results

The average character accuracies for non-stopwords for each generation are shown in Table 2. Table 2 shows that the average character accuracy of all non-stopwords from the MANICURE output is slightly better than for the raw OCR output. This improvement is more profound for higher generation photocopies (i.e., 97.79% for raw OCR and 98.06% for MANICURE for the fourth generation copy). In general, this result shows that the MANICURE post-processing system does improve the character accuracy of non-stopwords.

The results shown in Table 3 are even more significant. The word accuracy improvement of MANICURE output over raw OCR output for all non-stopwords ranges from 0.51% for generation 0 to 2.23% for generation 4. A one percent improvement

Table 3: Average Non-stopword Accuracy for 17 Document Sample

| Generation | RAW OCR | MANICURE |
|------------|---------|----------|
| Orig. | 97.44 | 98.01 |
| Gen 0 | 97.03 | 97.54 |
| Gen 1 | 96.45 | 97.23 |
| Gen 2 | 95.05 | 96.15 |
| Gen 3 | 92.78 | 94.64 |
| Gen 4 | 91.91 | 94.14 |

Table 4: Percentage of Incorrect Words Corrected by MANICURE

| Generation | Percentage Improvement | Percentage Corrected |
|------------|------------------------|----------------------|
| Orig. | 0.57 | |
| Gen 0 | 0.51 | 17.2 |
| Gen 1 | 0.78 | 22.0 |
| Gen 2 | 1.10 | 22.2 |
| Gen 3 | 1.86 | 25.8 |
| Gen 4 | 2.23 | 27.5 |

in this accuracy measure is very significant. Table 4 shows just how significant these improvements can be for non-stopword correction.

Combined, these results show that the improvement provided by MANICURE *increases as page quality decreases* revealing MANICURE's ability to correct OCR errors exceptionally well on low quality documents. This is exactly what we expected, since MANICURE was designed to improve the accuracy of non-stopwords in OCR output for the specific purpose of improving document retrievability. Overall, using non-stopword accuracy metrics, these results indicate that the character accuracies produced by the current DOE document conversion system are better than one could expect from raw unprocessed OCR text. Finally, because retrievability of documents from the LSN will be the primary use of the text produced, it seems clear that a test measuring retrieval effectiveness is at least as important. Section 4 discusses the retrievability tests conducted by ISRI for DOE's document conversion system.

4 Retrieval Test

The retrievability test [13] was designed to address the quality of retrieval of documents produced by the DOE for the LSN. Our experiment utilized the LSN IR technology to compare documents with nearly 100% accuracy to an identical document set prepared using DOE's conversion system. As mentioned in Section 3.1, building a large collection of OCR ground-truth data is an arduous task. The re-

Table 5: LSS Prototype Collection Statistics

| | |
|----------------------------------|--------|
| Document count | 1058 |
| Number of pages | 46,731 |
| Query Count | 62 |
| Avg # of relevant docs/query | 17 |
| Median # of relevant docs/query | 9 |
| Fewest relevant docs for a query | 1 |
| Most relevant docs for a query | 99 |

quirement for exact duplication in ASCII of an image page is difficult to obtain for even a small set of pages. Fortunately, ISRI has in its collections a 99.8% correct set of documents with queries and relevancy judgments that had been prepared for the LSS Prototype[7].³ The collection consists of 1058 documents, 62 queries, and 1104 relevancy judgments. This collection is particularly well-suited for determining how character accuracy may affect retrieval performance because of its similar characteristics and topic content to the planned LSN. We prepared another version that had been recognized and post-processed using the current DOE document conversion system. Collection statistics appear in Table 5. This collection should not be considered equivalent to OCR ground-truth since carriage-returns, spacing, columnization and other formatting features are not identical to the hard copy page. Still, the typed text was measured to be 99.8% correct. By comparing the results of the automatically recognized version to this 99.8% correct version, we could clarify whether or not improved character accuracy improved retrieval over what was produced from the DOE document conversion system.

We report on retrieval results using the standard measures of recall and precision[5]. We loaded and indexed both collections into the LSN IR system and the 62 queries were batch run against these two datasets in exactly the same way. The recall/precision results appear in Table 6.

Note that the difference between average precision for the two runs is less than 0.3%. This difference is too small to be considered statistically significant telling us that the process used by DOE to prepare the documents for the LSN will return results equivalent to a collection that was corrected to meet 99.8% character accuracy. With respect to retrievability then, an artificially high character accuracy does not necessarily guarantee better results for the end user.

³The LSS or Licensing Support System was a precursor to the LSN and the prototype was used to test its feasibility.

Table 6: 11-Point Precision for 99.8% Correct Text vs. Automatically Recognized and MANICURED Text

| Recall Points | 99.8% Correct | Auto-recognized w/MANICURE |
|---------------|---------------|----------------------------|
| 0.00 | 0.55 | 0.54 |
| 0.10 | 0.46 | 0.45 |
| 0.20 | 0.35 | 0.34 |
| 0.30 | 0.29 | 0.30 |
| 0.40 | 0.26 | 0.26 |
| 0.50 | 0.22 | 0.22 |
| 0.60 | 0.18 | 0.18 |
| 0.70 | 0.14 | 0.14 |
| 0.80 | 0.12 | 0.11 |
| 0.90 | 0.08 | 0.07 |
| 1.00 | 0.06 | 0.05 |
| Avg | 0.245 | 0.242 |

5 Conclusion and Summary

The objective of these tests was to evaluate the DOE document conversion system in a way that was compatible to its application but also meaningful for the technologies that make up the system. Character accuracy is an important metric when OCR technology is being evaluated. We have shown though that a more pertinent measure is non-stopword accuracy when the OCR text will be used for retrieval. Further, no measure of retrievability would be complete unless recall and precision are used to verify result set relevance. Both these tests were exercised in our DOE system evaluation.

There are several significant points implied by these tests:

- Non-stopword accuracy is a more appropriate metric for OCR-generated text when the text will be loaded into an IR system for retrieval.
- Applying document level post-processing, like MANICURE, allows for correction unattainable by an OCR device that applies page level recognition.
- MANICURE improves character accuracies on words that matter most for retrieval (i.e., non-stopwords).
- Retrievability from automatically generated OCR followed by MANICURE is equivalent to retrievability from 99.8% correct collections.
- These tests have demonstrated that the quality of the documents delivered by the DOE will give effective retrieval results for the users of the LSN.

References

- [1] Junichi Kanai, Thomas A. Nartker, Stephen V. Rice, and George Nagy. Performance metrics for document understanding systems. In *Proc. 2nd Intl. Conf. on Document Analysis and Recognition*, pages 424–427, Tsukuba Science City, Japan, October 1993.
- [2] T. A. Nartker. Need for information metrics: with examples from document analysis. In *SPIE Proceedings, Document Recognition*, pages 184–193, San Jose, CA, February 1994.
- [3] T. A. Nartker, R. B. Bradford, and B. A. Cerny. A preliminary report on UNLV/GT1: A database for ground-truth testing in document analysis and character recognition. In *Proc. 1st Symp. on Document Analysis and Information Retrieval*, pages 300–315, Las Vegas, NV, March 1992.
- [4] Tom Nartker and Ron Young. OCR accuracy produced by the current DOE document conversion system. Technical Report 2002-06, Information Science Research Institute, University of Nevada, Las Vegas, April 2002.
- [5] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw Hill, New York, 1983.
- [6] Scansoft, Inc., Peabody, MA. *Recognition API Manual*, v10 edition, 2000.
- [7] Science Applications International Corporation. Capture station simulation lessons learned. Final report for the Licensing Support System, 1990. Final report for the Licensing Support System prepared under contract DE-AC01-87RW00084 for the U.S. Department of Energy, Office of Civilian Radioactive Waste Management, Washington D.C.
- [8] Kazem Taghva, Julie Borsack, Bryan Bullard, and Allen Condit. Post-editing through approximation and global correction. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(6):911–923, 1995.
- [9] Kazem Taghva, Julie Borsack, and Allen Condit. Results of applying probabilistic IR to OCR text. In *Proc. 17th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 202–211, Dublin, Ireland, July 1994.
- [10] Kazem Taghva, Julie Borsack, and Allen Condit. Effects of OCR errors on ranking and feedback using the vector space model. *Inf. Proc. and Management*, 32(3):317–327, 1996.
- [11] Kazem Taghva, Julie Borsack, and Allen Condit. Evaluation of model-based retrieval effectiveness with OCR text. *ACM Transactions on Information Systems*, 14(1):64–93, January 1996.
- [12] Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. The effects of noisy data on text retrieval. *J. American Soc. for Inf. Sci.*, 45(1):50–58, January 1994.
- [13] Kazem Taghva, Julie Borsack, Steve Lumos, and Allen Condit. Retrievability of documents produced by the current DOE document conversion system. Technical Report 2002-05, Information Science Research Institute, University of Nevada, Las Vegas, April 2002.
- [14] Kazem Taghva, Allen Condit, Julie Borsack, John Kilburg, Changshi Wu, and Jeff Gilbreth. The MANICURE document processing system. In *Proc. IS&T/SPIE 1998 Intl. Symp. on Electronic Imaging Science and Technology*, San Jose, CA, January 1998.
- [15] Kazem Taghva and Jeff Gilbreth. Recognizing acronyms and their definitions. Technical Report 94-07, Information Science Research Institute, University of Nevada, Las Vegas, November 1994.
- [16] Kazem Taghva, Tom Nartker, Allen Condit, and Julie Borsack. Automatic removal of “garbage strings” in ocr text: An implementation. In *The 5th World Multi-Conference on Systemics, Cybernetics and Informatics*, Orlando, Florida, July 2001.

Varying Effects of Image Improvement Methods on OCR Accuracy

Kristen Summers
Vredenburg
4831 Walden Lane
Lanham, MD 20706
ksummers@vredenburg.com

Abstract

Optical Character Recognition (OCR) software is widely available and generally performs very well on extremely clean document images, but its accuracy can degrade significantly with the introduction of image noise, even when this noise appears minor to human eyes. A great number of methods exist to "clean" images with particular types of noise (such as despeckling, filling broken lines, etc.); when applied to images that exhibit the problems they correct, these methods can substantially improve OCR results on those images. When applied to some images, however, these same methods degrade the images and decrease OCR accuracy. In the process of developing a trained system to select methods to apply, we have collected OCR scores (defined as the average of recall and precision) for a corpus of 540 documents from the Congressional hearings on tobacco [1], in the original forms and after applying each of 9 different image transformations. We describe the varying effects of these transformations.

1 Introduction

Inspired by [2] and [3], we have been developing a trained system that learns to associate image characteristics with preferred transformations that attempt to improve the images. Since we define the value of a transformation by its effect on OCR results, we have collected OCR scores (defined as the average of recall and precision) for a corpus of 540 documents from the Congressional hearings on tobacco, in the original forms and after applying each of 9 different image transformations. The tobacco collection includes images of highly varying quality, from different sources, and spanning several decades, so it provides a good basis for experimenting with the effects of image transformations on a heterogeneous collection of documents.

2 Related Work

The current work is inspired by the QUARC system [2] developed at Los Alamos National Labs and also by earlier work described in Ref. [3]. Both of these previous systems define a set of image characteristics to measure and a set of image transformation methods to consider. They then use machine learning, construing the problem as one of supervised classification, with one class for each transformation method and one class for retaining the original image.

The input for a document is its set of characteristic values, and the class is the transformation method that yields the largest possible improvement in OCR results. If no method improves the OCR, the correct class is the one that corresponds to applying no method, i.e., using the original image. Training thus proceeds on a set of documents for which OCR ground truth is known. Each transformation is applied in turn; OCR is performed on each transformed image, as well as the original; and all these results are evaluated, in order to determine the correct class. Once the system has generalized over the training data, application of the learned behavior to a new (or test) document image is straightforward. The characteristics are measured, the classification method is applied to this input, and the resulting class determines a single transformation method (or none), which is then applied to the image.

We have been developing a system, called ImageRefiner, that follows a similar general approach; it combines more general image characteristics with those of QUARC and is intended to apply to a broader range of documents. It also considers whether to apply each transformation as a separate question, rather than focusing entirely on selecting a single best transformation. This system provides the context for the examination of the effects of different image transformations on OCR accuracy.

Additional related work includes both development of image transformations that are likely to

remove particular kinds of noise, such as those described in Refs. [4–6], and analysis of images to determine the types and sources of noise present, as in Refs. [7–9].

3 Image Transformations

ImageRefiner currently includes the 9 different QUARC transformation methods that can be used without special requirements for the page image, i.e., those that do not require imposing a “typewriter grid” on the page. These methods are: application of a morphological filter intended to fill character holes and breaks, a moderate despeckling filter, an aggressive despeckling filter, morphological closing, morphological opening, the kFill filter for salt-and-pepper noise, with its grid set to 3×3 , kFill with the grid set to 4×4 , kFill with the grid set to 5×5 , and a “depebbling” procedure.

Fill holes and breaks. This transformation applies filters designed to fill small holes by performing erosion, then applying OR.

Moderate Despeckle. This is a white 3×3 dilation.

Aggressive Despeckle. This is performed by doing white dilations with two different 5×5 filters, each of which is a full 3×3 , with 2 extra “nubbins.” The output is the “AND” of these results.

Morphological Open. This is a simple 3×3 opening.

Morphological Close. This is a simple 3×3 closing.

Kfill 3×3 , 4×4 , and 5×5 . This repeatedly applies (for black and for white) a filter that may reset interior pixels, based on: the count of the pixels of the color of interest, the count of corner pixels of that color, the count of connected components of that color.

Depebble. This removes small black connected components. The minimum size for retained components is based on a single calculated font size for the image.

4 Data Set

We collected the effects of each transformation on OCR accuracy for each of 540 documents from the congressional hearings on tobacco. The tobacco collection includes documents from a wide variety of time periods and original sources, and it naturally includes images of widely varying quality. It thus provides excellent material for experimenting with

OCR. Sample document fragments are shown in Figure 1.

In order to evaluate OCR accuracy on the documents, ground truth text was entered by hand. This ground truth treats the entire page as a single zone, and we ran the OCR software in the same way. The ground truth entry matches the physical appearance of the page as closely as possible. For instance, consider a stamp on a document. If a group of characters from the stamp align with a line of printed text on the page, the ground truth file includes the stamp characters on that same line (in the order in which they appear). If a group of stamped characters appears between the lines formed by the printed text of the page, these stamped characters appear in the ground truth as a separate line, between the surrounding lines of printed text. Treating the entire page as one zone in this way serves our purpose of focusing on differences in *character* recognition accuracy, rather than the segmentation of pages into zones.

5 Effects of Transformations

In order to measure the effects of the various transformations on OCR accuracy, we performed each transformation on each of the images in our collection and performed OCR on the resulting transformed images, as well as the original images. We then evaluated these OCR results against the ground truth described in Section 4.

5.1 OCR Accuracy Evaluation

To evaluate the OCR accuracy on images, we first matched the ground truth text with the OCR output text using a common line-based string edit distance comparison[10–12]. This approach uses the standard string edit distance that calculates the cost of transforming one string into another, based on costs of inserting, deleting, and changing individual items [13]. The edit distance is applied to the lines of text in the OCR output and in the ground truth, with the cost of operating on a line defined as the character-based cost of the same operation (i.e., the cost of inserting a line is the cost of inserting the number of characters it contains, the cost of deleting a line is the cost of deleting the number of characters it contains, and the cost of changing one line into another is the character-based string edit distance between the text lines). The character-based transformation path that yields the minimum cost (i.e., the edit distance) provides the basis for scoring the OCR output; characters that are unchanged in this path are matched.

Using this basis for matching, we calculated character-level precision (percentage of OCR output characters with matching characters in the ground

TO: E. Papples
 FROM: J. K. Wells, III
 Corporate Counsel

Compromised Individua

Comprehensive Effectual Summary of
 Various Experiments with Tobacco
 Smoke Conducted at Cancer Institute
 of James Alton

The attached abstract summarizes the
 epidemiological studies examining occu
 risk in smokers vs. non-smokers.

A study f

University of Califo

CHADBOURNE, PARKE, WHITESIDE & WOLFF
 30 ROCHEFELLER PLAZA, NEW YORK, N. Y. 10020
 212-541-5500

Brownson, R.C., Reif, J.S., Keefe,
 Pritzl, J.A., "Risk Factors for Ader
 American Journal of Epidemiology 125

SHOOK, HARDY, OTTMAN, MITCHELL & BACON

DAVID R. HANCOY
 JAMES H. DETHMERS
 EUGENE M. MITCHELL
 CHARLES L. BACON
 DAVID H. CLARK
 LANE D. BAUER
 FRANK P. DEORIE
 FREDERICK BECHTOLD
 WILLIAM W. SHERR
 JOHN G. HODGE
 DONALD W. MOEL
 EVERETT A. DOLAN, JR.
 JOHN T. MARTIN
 DAVID H. BROWN

615 GRAND AVENUE
 KANSAS CITY, MISSOURI 64108
 TELEPHONE 274-8889
 AREA CODE 913

FRANK A. SEBNE (1934-1940) SAM R. SEFFER (1938-1944) EDGAR SHOOK (1934-1940)

WILLIAM E. WAUGH, JR.
 RUSSELL S. NOBLETT
 LEE L. STANTFORD
 DAVID K. HARTY
 WILLIAM E. SHUMERMAN
 ROBERT L. NORTHROP
 PATRICK H. LEMLEY
 C. MATH LARSON
 JOHN C. MONICA
 CHARLES E. WALL
 HARVEY L. KAPLAN
 DAVID H. ROBERTS

May 21, 1971

Figure 1: Sample fragments from tobacco document images

truth) and recall (percentage of ground truth characters with matching characters in the OCR output), and we averaged these two measures to find a single score in the range [0, 1].

5.2 Effects on OCR Accuracy

We measure the effect of a transformation as the *difference* in OCR score between the result for the transformed image and the result for the original image. We do not use the percentage of error reduction, since for images with good scores, even very small improvements will lead to reductions of a large percentage of the error, since the original error itself is so small. We consider all differences in the range $[-.01, .01]$ to be negligible, as we have observed non-determinism in the OCR process that can affect the score by about .01; we have no guarantee that it cannot affect the score by a larger amount, but we have not observed this behavior. We quantify other differences into the ranges $[.2, 1]$, $[.15, .2]$, $[.1, .15]$, $[.05, .1]$, $(.01, .05)$, $[-.05, -.01]$, $[-.1, -.05]$, $[-.15, -.1]$, $[-.2, -.15]$, and $[-1, -.2]$.

Figures 2-6 show histograms of the number of documents with OCR accuracy differences that fell into each range for the various transformations. All transformations damaged OCR accuracy more often than they improved it, and with a small number of exceptions, larger changes (whether positive or negative) were less frequent than smaller changes.

For all transformations other than despeckling, the most common effect was a difference with an absolute value of less than .01. Both moderate and aggressive despeckling, however, decreased the OCR

scores by at least .2 on a greater number of documents than the number on which they made an impact of less than .01; these transformations are the likeliest to cause damage in this set of images. (It was easier to learn whether to apply these transformations than some of the others, however, so the fact that they often cause damage does not mean that they should not be considered.)

Table 1 summarizes the effects of the transformations on those images for which they improve OCR accuracy by at least .01. It shows the number of images for which each transformation made such an improvement, the maximum improvement made by each transformation, and the average and standard deviation for each. Table 2 shows the same information for degradations in OCR accuracy of at least .01.

The despeckling procedures may be the most dangerous, in that they are the most likely to cause a decrease in accuracy, but they also have the highest average improvements on those images for which they are helpful. The despeckling procedures, Morphological Open, and Kfill 3×3 all made an improvement of over .9 on at least one image; this corresponds to taking the OCR output from essentially meaningless to nearly full accuracy. The other Kfill filters never improved an image by much more than .1. The despeckle procedures and Morphological Open all created a degradation of over .9 on at least one image; this corresponds to taking the OCR output from nearly full accuracy to essentially meaningless. Note that the depebling procedure is the only transformation that improved an image by over

Fill Holes and Breaks: OCR Accuracy Differences

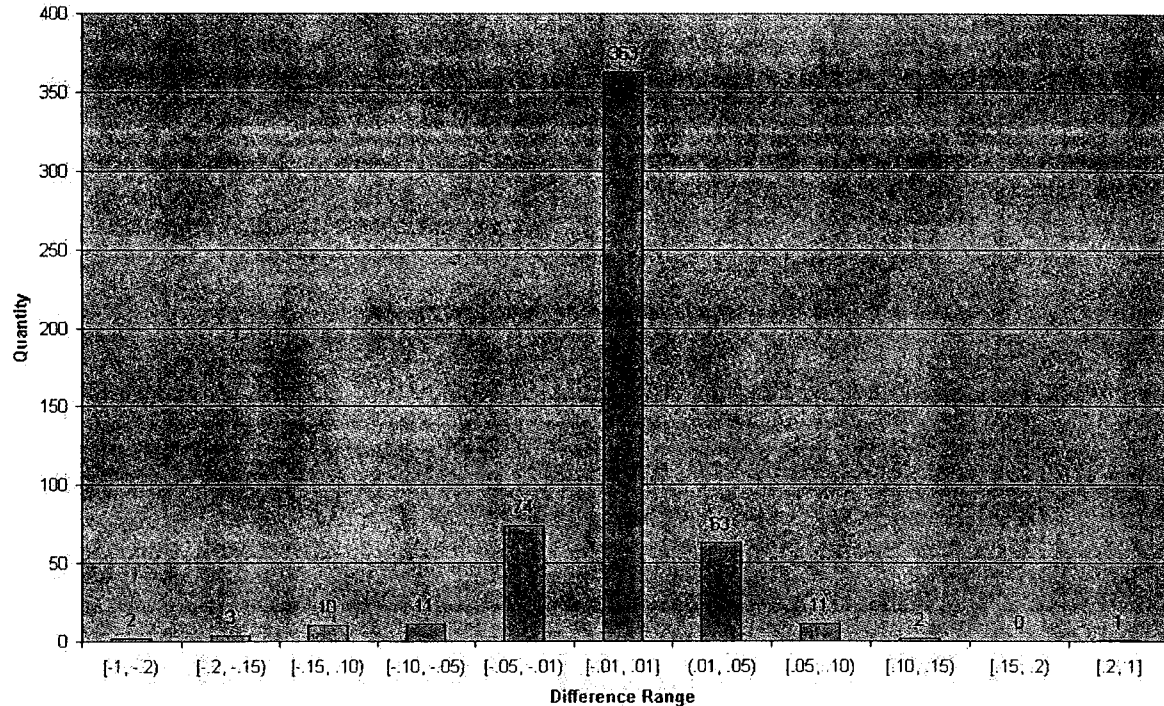


Figure 2: Histogram of effects on OCR accuracy of Fill Holes and Breaks.

.9 but did not degrade any images as substantially. Morphological Close degraded an image by over .7, but its best improvement was .325.

The same document yields the maximum improvement with Aggressive Despeckle and with Morphological Open. Figure 7 shows the original full image and the same image after morphological open.

Some effects of transformations that significantly improve OCR accuracy are not readily apparent to visual inspection. For example, consider the image in Figure 9. The OCR accuracy improves after Morphological Close by .236; this is the second largest improvement achieved by Morphological Close in our corpus. Figure 10 shows closer views of portions of this page before and after the transformation; the differences are still difficult to notice. Figure 11 shows the OCR output for the selected fragments.

6 The ImageRefiner Trained System

We performed an experiment for each of the 9 transformations. Each experiment addressed the problem of learning whether or not to perform its transformation. All the experiments used the same set of image characteristics, an expansion of those used in QUARC with several lower-level characteristics, and they used sets of 3 neural networks that participated

in a majority vote for the correct class.

Each experiment was performed with a subset of 150 documents from our corpus. Each subset was selected to include a mix of documents for which its transformation is helpful, documents for which it is harmful, and documents for which it makes very little difference (less than .01 effect on the OCR score). Otherwise, the selection was random. We performed 30-fold cross-validation on each set; that is, in 30 repeated runs, we held out 5 documents, trained on the remaining 145, and tested on the held-out 5, holding out a different 5 documents each time.

In training, documents on which the transformation has very little effect were considered examples of the “do not transform” class, since we prefer to avoid unnecessary work. However, the important goal of the training is to learn to take the correct action when it matters, and it is more important to take the correct action when the impact is greater. Considering all cases, the system made the correct choice 64% of the time, ranging from 43% to 75% for the individual transformations. In general, it performed better on documents in which the impact was greater; Table 3 shows different thresholds of score differences (positive or negative) due to the transformation, together with the number of images in the set for which

Despeckle: OCR Accuracy Differences

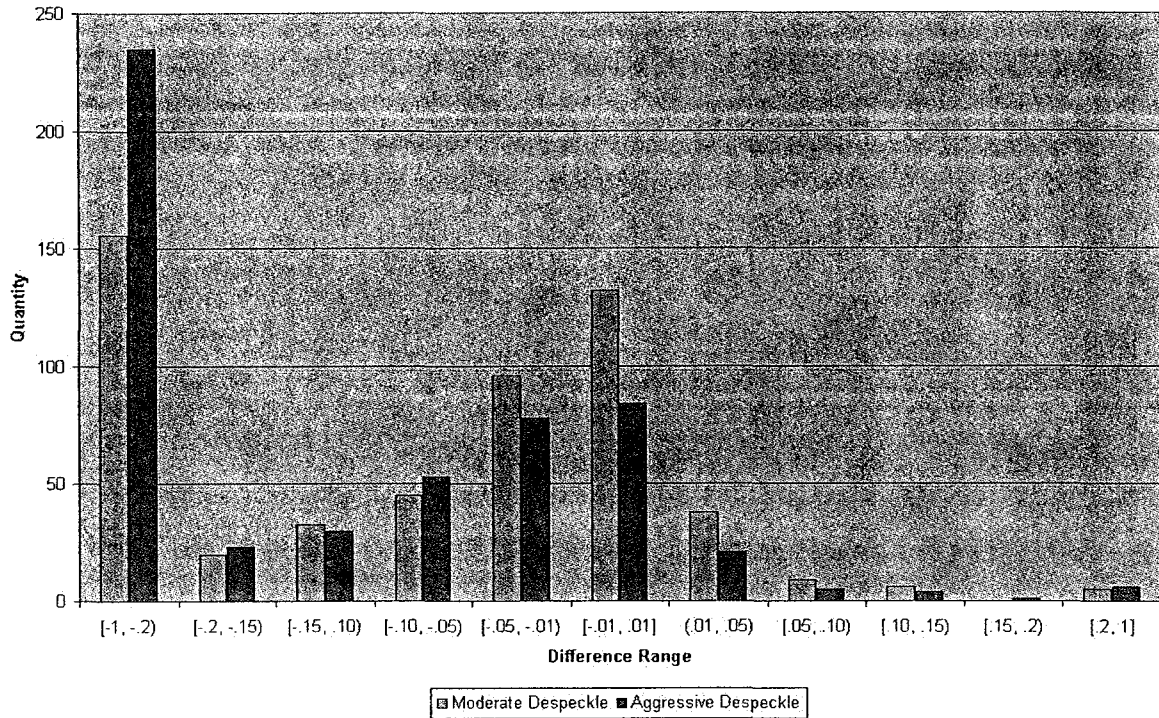


Figure 3: Histogram of effects on OCR accuracy of Moderate and Aggressive Despeckle.

Table 1: Improvements of at least .01

| Transform | Images Improved | Maximum | Average | Std. Dev. |
|-----------------------|-----------------|---------|---------|-----------|
| Fill Holes and Breaks | 77 | 0.256 | 0.035 | 0.036 |
| Moderate Despeckle | 58 | 0.982 | 0.0970 | .165 |
| Aggressive Despeckle | 37 | 0.967 | 0.113 | 0.185 |
| Morphological Close | 52 | 0.325 | 0.047 | 0.060 |
| Morphological Open | 56 | 0.983 | 0.066 | 0.139 |
| Kfill 3x3 | 58 | 0.965 | 0.053 | 0.129 |
| Kfill 4x4 | 41 | 0.105 | 0.025 | 0.020 |
| Kfill 5x5 | 22 | 0.101 | 0.032 | 0.027 |
| Depebble | 56 | 0.972 | 0.075 | 0.153 |

Table 2: Degradations of at least .01

| Transform | Images Degraded | Maximum | Average | Std. Dev. |
|-----------------------|-----------------|---------|---------|-----------|
| Fill Holes and Breaks | 100 | 0.287 | 0.047 | 0.052 |
| Moderate Despeckle | 350 | 0.997 | 0.313 | 0.338 |
| Aggressive Despeckle | 419 | 1.000 | 0.402 | 0.358 |
| Morphological Close | 158 | 0.788 | 0.057 | 0.079 |
| Morphological Open | 258 | 0.995 | 0.173 | 0.246 |
| Kfill 3x3 | 105 | 0.243 | 0.043 | 0.043 |
| Kfill 4x4 | 107 | 0.181 | 0.043 | 0.044 |
| Kfill 5x5 | 115 | 0.280 | 0.042 | 0.050 |
| Depebble | 151 | 0.237 | 0.045 | 0.045 |

Morphological Close/Open: OCR Accuracy Differences

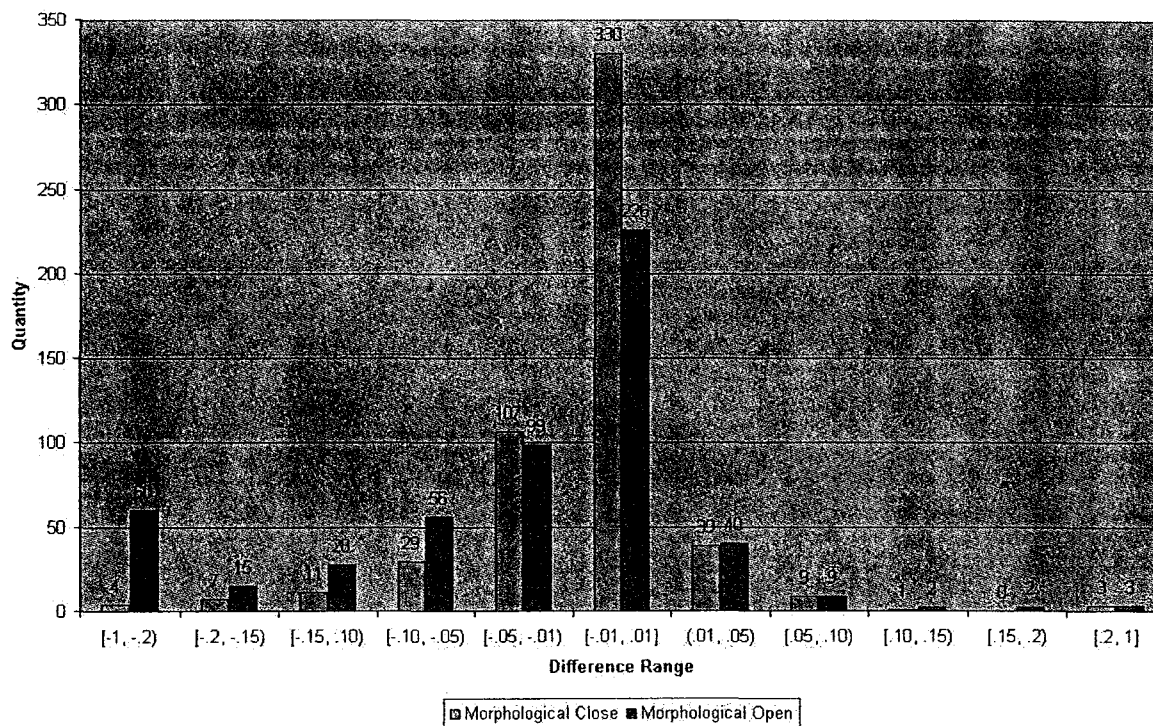


Figure 4: Histogram of effects on OCR accuracy of Morphological Close and Open.

the transformation impact exceeded this threshold and the percentage of such documents for which the system made the correct decision. Figure 6 graphs the percentages of correct decisions for the sets with impacts exceeding the various thresholds. On those documents for which the decision made a difference in score of at least .05, the system made the correct choice 76.6% of the time.

The small number of documents for which many of these transformations affect the OCR score substantially suggests both that other transformations may be better suited to this corpus and that applying multiple transformations is likely to have a stronger effect.

7 Conclusions and Future Work

The differences in effect of the same transformation on different images confirms the need to select carefully the image improvement methods to apply in any given instance. The fact that the impact of an image transformation on OCR accuracy may not correspond to what is noticeable to a human reader supports the conclusion that selecting the method to use by visual inspection is not optimal in its accuracy, as well as often forming an unacceptable burden for the human operator. The fact that the par-

ticular transformations we used so often had negligible effects on the accuracy of OCR on the images that we considered may indicate a poor match between the transformations and the images, or it may indicate that the cases are comparatively rare in which a single application of a method of this type makes a dramatic difference in OCR accuracy.

As we continue to pursue automatic methods of selecting transformations to apply to images, we intend to investigate whether a different choice of candidate transformations shows different results. We also intend to expand this work to consider sequences of transformations; we expect to create an iterative application that repeatedly selects transformations to apply, since many images may be best served by a combination of transformations. The order in which the system applies transformations can be critical; a transformation can be harmful to an original image but helpful to the version of that image created by another transformation. For example, Figure 13 shows a portion of an original image that is substantially improved by morphological closure and degraded by moderate despeckling; the result of morphological closure is further improved by an application of moderate despeckling. Figures 14-16 show the results of these transformations.

Kfill: OCR Accuracy Differences

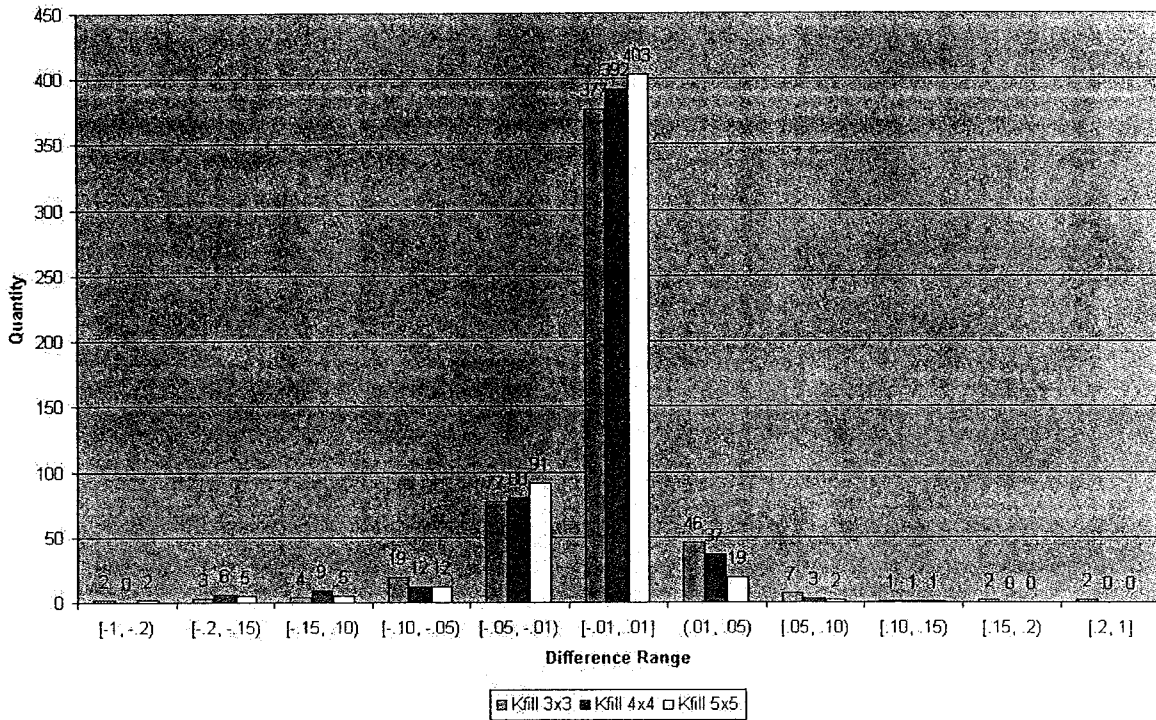


Figure 5: Histogram of effects on OCR accuracy of Kfill.

Table 3: Transformation methods and thresholds.

| Method | | Threshold | | | | | |
|-----------------------|-------------------|-----------|-----|-----|-----|-----|-----|
| | | 0 | .01 | .02 | .03 | .04 | .05 |
| Fill holes and breaks | Correct Decisions | 62% | 63% | 71% | 78% | 80% | 80% |
| | Number of images | 150 | 120 | 70 | 46 | 35 | 30 |
| Moderate despeckle | Correct Decisions | 67% | 69% | 70% | 72% | 74% | 75% |
| | Number of images | 150 | 100 | 77 | 71 | 66 | 57 |
| Aggressive despeckle | Correct Decisions | 75% | 82% | 85% | 87% | 87% | 88% |
| | Number of images | 150 | 110 | 100 | 94 | 87 | 83 |
| Morphological close | Correct Decisions | 60% | 66% | 74% | 78% | 78% | 78% |
| | Number of images | 150 | 100 | 69 | 50 | 37 | 32 |
| Morphological open | Correct Decisions | 63% | 65% | 73% | 71% | 75% | 75% |
| | Number of images | 150 | 100 | 71 | 56 | 51 | 48 |
| KFill 3 × 3 | Correct Decisions | 49% | 45% | 50% | 51% | 60% | 61% |
| | Number of images | 150 | 100 | 58 | 37 | 25 | 23 |
| KFill 4 × 4 | Correct Decisions | 66% | 69% | 68% | 79% | 81% | 75% |
| | Number of images | 150 | 75 | 41 | 19 | 16 | 12 |
| KFill 5 × 5 | Correct Decisions | 73% | 74% | 71% | 67% | 50% | 38% |
| | Number of images | 150 | 65 | 34 | 21 | 12 | 8 |
| Depebble | Correct Decisions | 65% | 69% | 73% | 71% | 71% | 69% |
| | Number of images | 150 | 100 | 73 | 52 | 38 | 32 |

Depebble: OCR Accuracy Differences

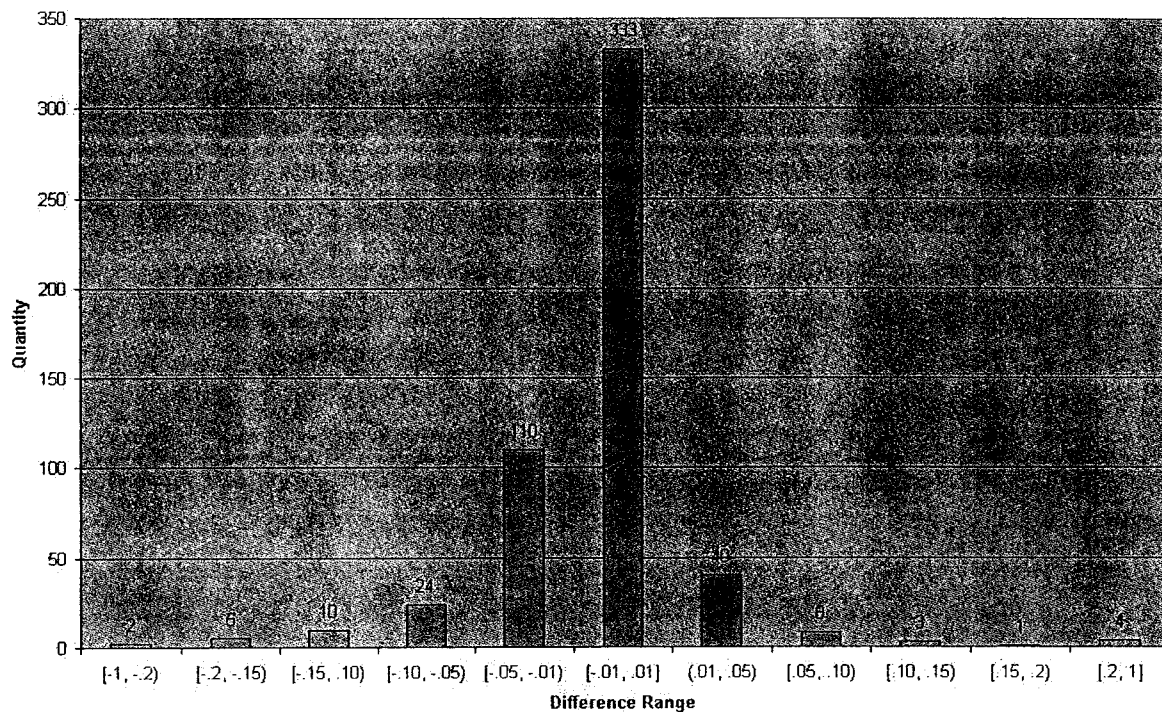


Figure 6: Histogram of effects on OCR accuracy of Depebbling.

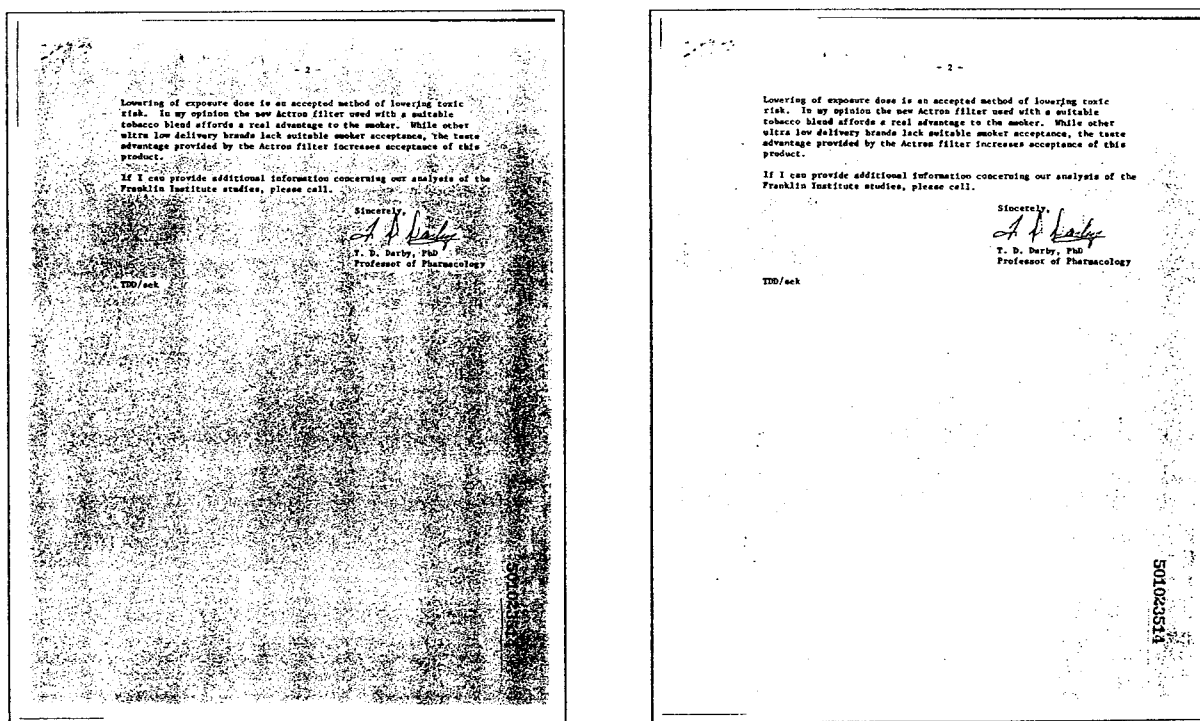


Figure 7: The original image (left) that is most improved by Aggressive Despeckle and by Morphological Open, and the same image after Morphological Open (right).

Original

Lowering of exposure dose is an accepted method of lowering toxic risk. In my opinion the new Actron filter used with a suitable tobacco blend affords a real advantage to the smoker. While other ultra low delivery brands lack suitable smoker acceptance, the taste advantage provided by the Actron filter increases acceptance of this product.

Morphological Open

Lowering of exposure dose is an accepted method of lowering toxic risk. In my opinion the new Actron filter used with a suitable tobacco blend affords a real advantage to the smoker. While other ultra low delivery brands lack suitable smoker acceptance, the taste advantage provided by the Actron filter increases acceptance of this product.

Aggressive Despeckle

Lowering of exposure dose is an accepted method of lowering toxic risk. In my opinion the new Actron filter used with a suitable tobacco blend affords a real advantage to the smoker. While other ultra low delivery brands lack suitable smoker acceptance, the taste advantage provided by the Actron filter increases acceptance of this product.

Figure 8: A portion of the image in Figure 7, in its original form, after morphological open, and after aggressive despeckle.

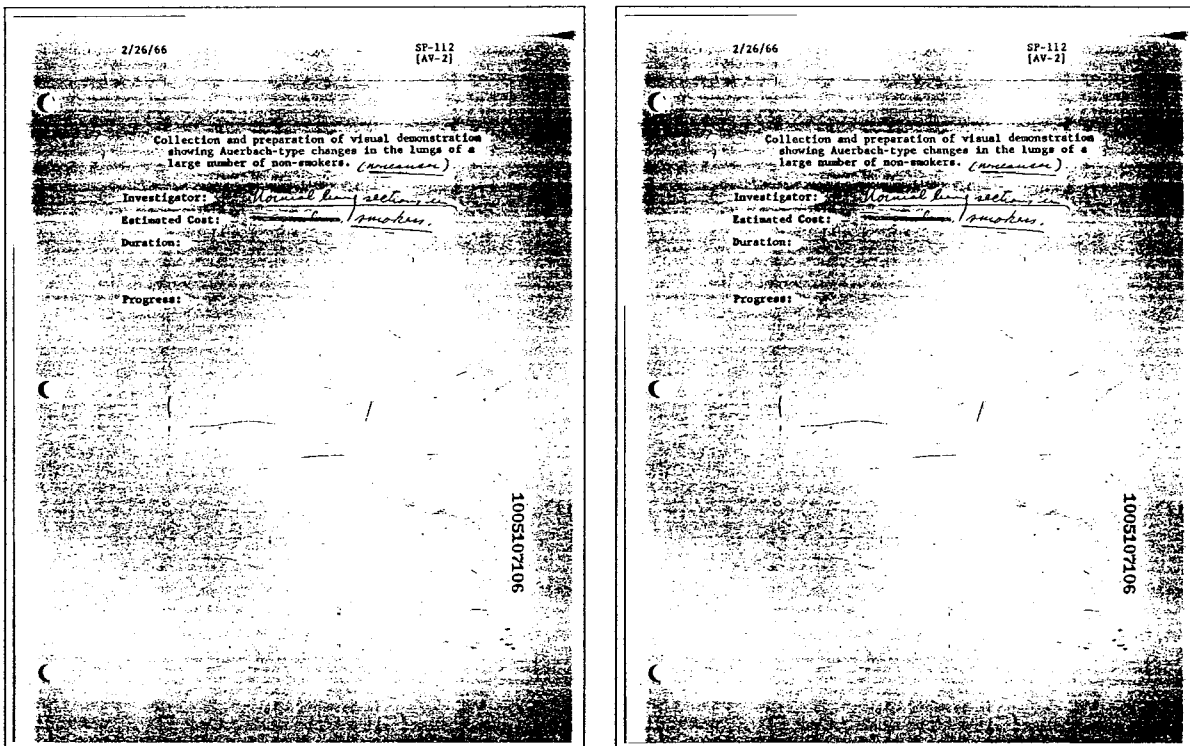


Figure 9: An original page and the same page after applying a morphological close operation that substantially improves the OCR results.

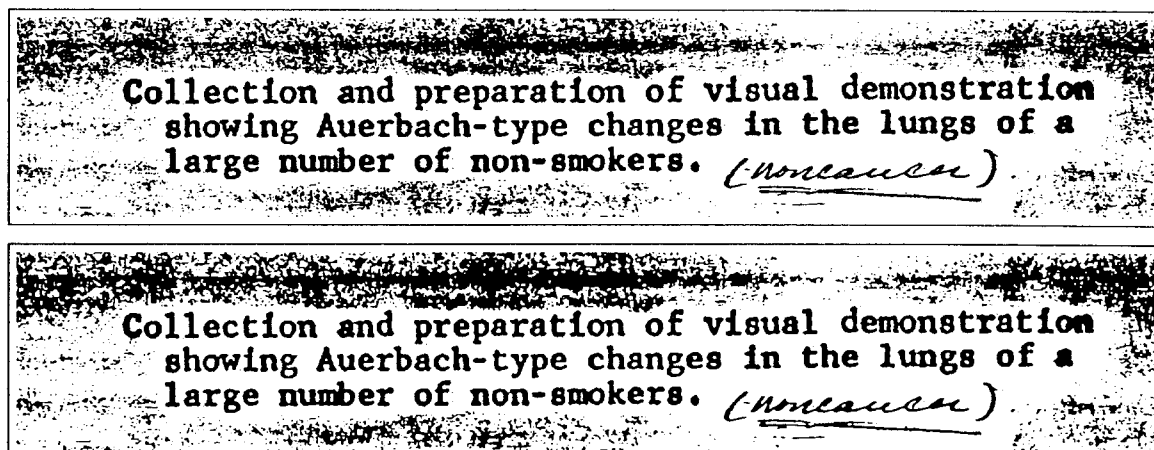


Figure 10: Enlarged portions of the original (upper) and transformed (lower) images in Figure 9.

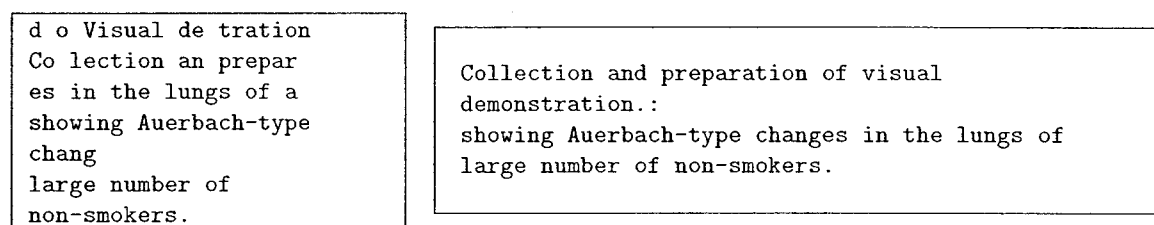


Figure 11: OCR outputs corresponding to the portions of the original image (left) and transformed (right) of Figure 9 that are expanded in Figure 10. After the morphological close, the text lines are correctly identified; additionally, some individual characters are recognized after transformation that are missed without it.

References

- [1] Committee on commerce tobacco documents. <http://www.house.gov/commerce/TobaccoDocs/documents.html>.
- [2] M. Cannon, J. Hochberg, and P. Kelly. Quality assessment and restoration of typewritten document images. *International Journal of Document Analysis and Recognition*, 2(2-3), 1999.
- [3] J. McNamara, D. Casey, R. Smith, and D. Bradburn. A classifier for evaluating the effects of image processing on character recognition. In *Proceedings: Image Algebra and Morphological Image Processing IV*, pages 109–120, 1993.
- [4] R. Loce and E. Dougherty. *Enhancement and Restoration of Digital Documents: Statistical Design of Nonlinear Algorithms*. SPIE-The International Society for Optical Engineering, Bellingham, 1997.
- [5] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Van Nostrand Reinhold, New York, 1994.
- [6] J. Hobby and T. Ho. Enhancing degraded document images via bitmap clustering and averaging. In *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, pages 394–400, 1997.
- [7] H. Baird. The state of the art of document image degradation modeling. In *Proceedings of the Workshop on Document Analysis Systems*, 2000. Available: <http://www.parc.xerox.com/istl/members/baird>.
- [8] L. Blando, J. Kanai, T. Nartker, and J. Gonzalez. Prediction of OCR accuracy. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 319–322, 1995.
- [9] G. Nagy, T. Nartker, and S. Rice. Optical character recognition: An illustrated guide to the frontier. In *Proceedings: Document Recognition and Retrieval VII*, pages 58–69, 2000.
- [10] Su Chen, M. Y. Jaisimha, Jaekyu Ha, Robert M. Haralick, and Ihsin T. Phillips. Reference manual for UW English document image database I: Version 1.2. Available on CD from the University of Washington, August 1993.

Results at Thresholds

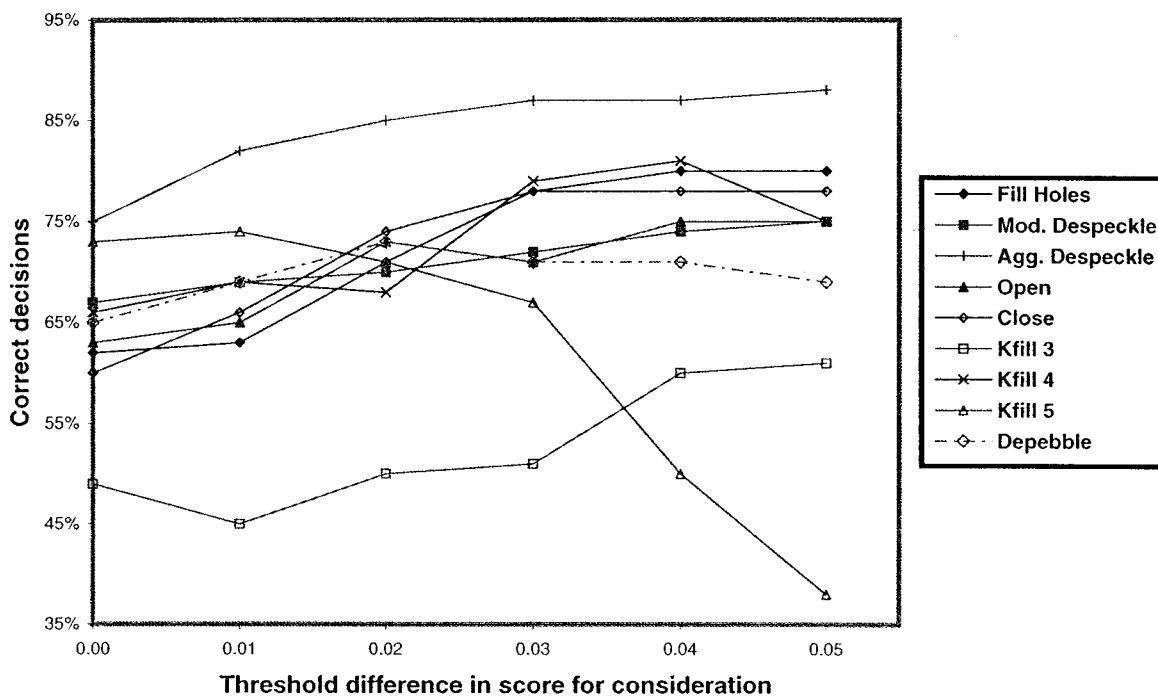


Figure 12: Correct decisions out of those with effects of at least the value of each threshold.

- [11] J. Esakov, D. Lopresti, and J. Sandberg. Classification and distribution of optical character recognition errors. In *Proceedings: Document Recognition*, pages 204–216, 1994.
- [12] K. Summers. OCR accuracy of three systems on English and Russian documents of highly varying quality. In *Proceedings: 2001 Symposium on Document Image Understanding Technology*, pages 123–128, 2001.
- [13] D. Sankoff and J. Kruskal. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, 1983.

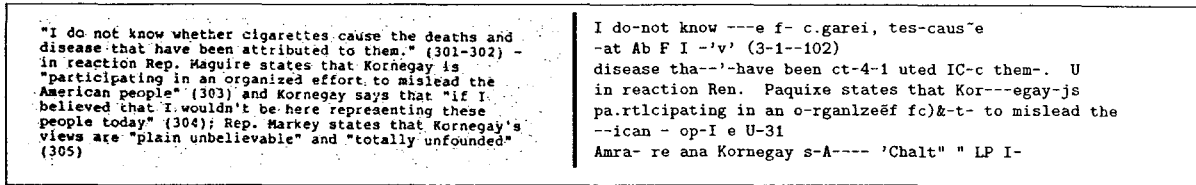


Figure 13: A fragment of the original version of an image, with the start of the OCR output that corresponds to this fragment.

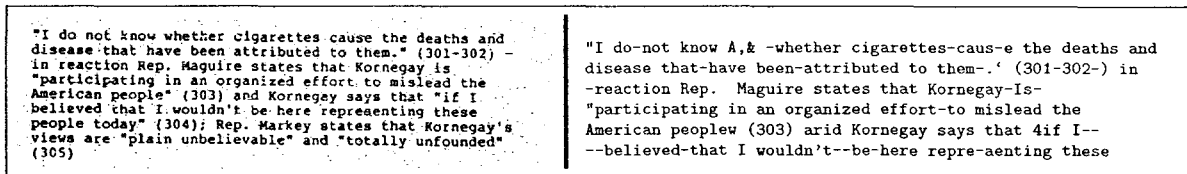


Figure 14: The image fragment in Figure 13 after a morphological close operation, with the start of the OCR output that corresponds to this fragment.

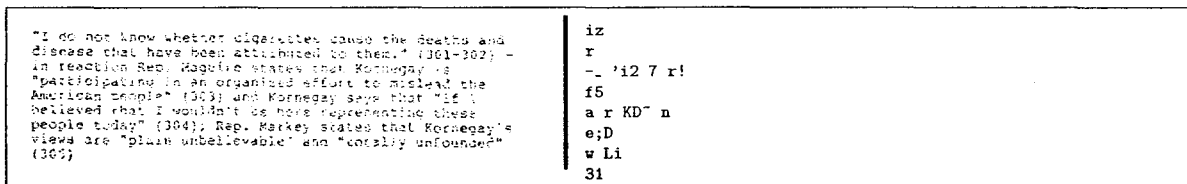


Figure 15: The image fragment in Figure 13 after a moderate despeckling operation, with several sample OCR output lines. In this case, no portion of the page OCR output is recognizable as corresponding to the selected image fragment.

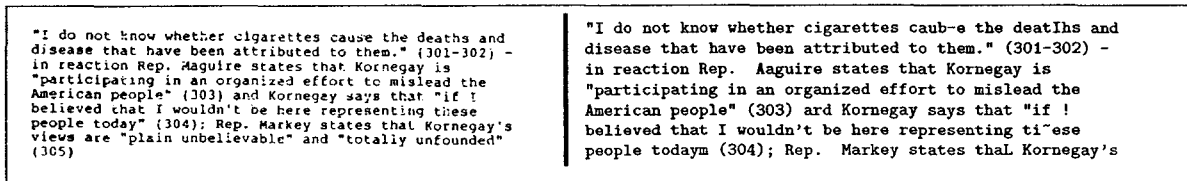


Figure 16: The image fragment in Figure 13 after morphological close (Figure 14) followed by moderate despeckle, with the start of the OCR output that corresponds to this fragment.

OCR Correction Using Historical Relationships from Verified Text in Biomedical Citations

Susan Hauser, Tehseen Sabir, George Thoma

Lister Hill National Center for Biomedical Communications
National Library of Medicine
National Institutes of Health
Department of Health and Human Services
Bethesda, Maryland 20894

Abstract

The Lister Hill National Center for Biomedical Communications has developed a system that incorporates OCR and automated recognition and reformatting algorithms to extract bibliographic citation data from scanned biomedical journal articles to populate the NLM's MEDLINE® database. The multi-engine OCR server incorporated in the system performs well in general, but fares less well with text printed in the small or italic fonts often used to print institutional affiliations. Because of poor OCR and other reasons, the resulting affiliation field frequently requires a disproportionate amount of time to manually correct and verify. In contrast, author names are usually printed in large, normal fonts that are correctly recognized by the OCR system. We describe techniques to exploit the more successful OCR conversion of author names to help find the correct affiliations from MEDLINE data.

1 Background

The Medical Article Records System, MARS, was developed by the Lister Hill National Center for Biomedical Communications to semi-automatically generate electronic bibliographic records from paper-based journal articles for the National Library of Medicine's MEDLINE® database [1] [2]. The system incorporates a commercial OCR server to convert the scanned page and several in-house developed modules to automatically construct the record from the OCR output text [3] [4]. Human operators verify and correct the automatically extracted fields, and type in those fields that are not automatically generated.

For several reasons, the affiliation field requires a disproportionate amount of time to correct and verify [5] [6]: 1) the affiliation field is often printed in small and/or italic fonts which are poorly converted by the OCR server; 2) the printed affiliation field frequently contains multiple affiliations, one for each author, while only the affiliation of the first author is included in the bibliographic record; 3) affiliations from the United States are to end with "USA", which frequently is not in the printed affiliation; 4) the OCR server does not recognize diacritical characters, which are common in non-USA affiliations, and because the standard keyboard does not include diacritical characters, the verification operator must insert individual diacritics by selecting from special "keys" on the monitor screen.

Figure 1 is typical of what the operator sees while verifying the affiliation field. The top of the screen displays part of the scanned image, while the bottom of the screen displays the OCR text for the field being corrected, with low-confidence characters highlighted in red. In this case, the operator must delete three affiliations, and correct several words in the first affiliation, the only one to be retained by MEDLINE's conventions.

Figure 1 also illustrates the usual case, where author names are printed in a large, regular font. The OCR server correctly converts characters printed in large, non-italic fonts, and numerals printed in any font. The one zip code that does appear in the affiliation is correctly converted, even though it is printed in italics.

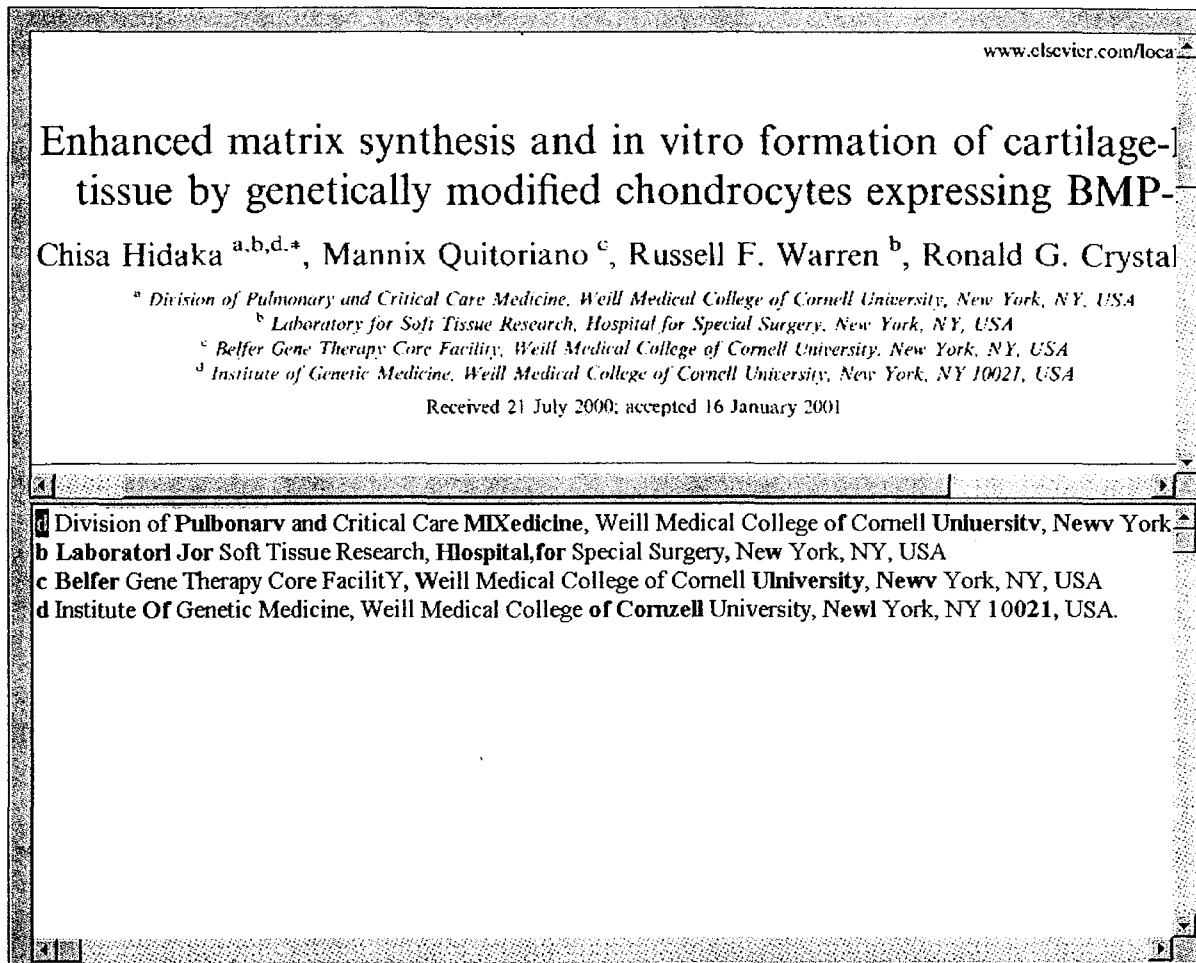


Figure 1. The verification operator's view of the OCR text from an affiliation field (bottom panel) and the corresponding image (top).

The MEDLINE database contains 12 million indexed citations for biomedical journal articles. Most of these citations include a list of one or more authors and the affiliation of the first author. Many authors publish repeatedly while at the same institution. Our objective is to use the historical author and affiliation relationships from this large dataset to find potentially correct, complete affiliations based on the author text and the affiliation text in the OCR output, and to present these affiliations to the verification operator in addition to the OCR text. The operator selects the affiliations as is, or edits them to create the correct affiliation field. Even if the affiliation presented is not structured exactly as the printed one, the operator may determine that it is easier, and more reliable, to edit the alternate affiliation than to correct the affiliation text from the OCR.

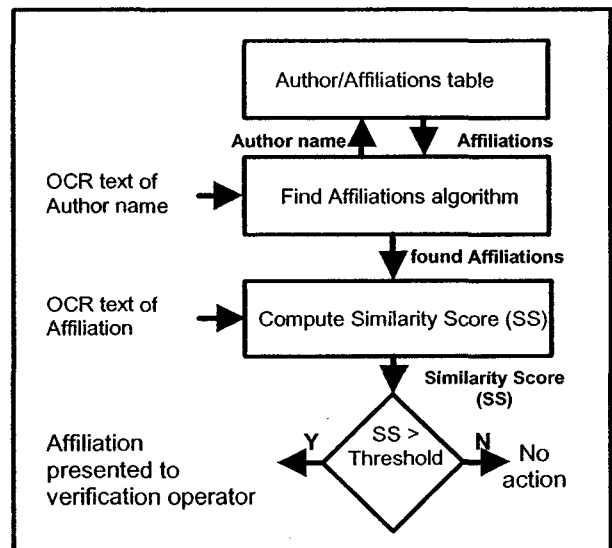


Figure 2. How the author/affiliation relationship is used.

Figure 2 shows how the module to find alternate affiliations is structured. The module uses the OCR text of the author name to query the author/affiliation table for potential affiliations. Each affiliation found in the table is compared to the OCR text of the affiliation to determine if it is the same institution as the OCR affiliation. If it is determined to be the same institution, it will be presented to the verification operator in addition to the OCR affiliation text.

2 Previous Work

A preliminary study was conducted with MARS output data, using a table of about 324,000 author/affiliation pairs extracted from MARS verified data, and a test set of about 20,000 first authors extracted from a separate set of MARS verified data [7]. 47% of the author names in the test set were found in the author/affiliation table. For 43% of these authors, at least one of the affiliations associated with the author name was the same institution as the institution in the OCR text. Thus we anticipate that a correct affiliation can be offered to the verification operator for at least 20% ($= 47\% \times 43\%$) of the articles that are processed.

Based on these encouraging data, we constructed a more complete database of author/affiliation relationships from MEDLINE. This table contains over 700,000 unique author/affiliation pairs taken from MEDLINE entries from 2000 through 2002.

Of the 394,768 unique author names in the author/affiliation table, about 34% are paired with more than one affiliation. Thirty five author names are paired with over 100 affiliations. Thus, even when an author name is found in the author/affiliations table, it is still necessary to determine which, if any, of the associated affiliations is the same one as that from the printed article. This is done by comparing the affiliations found in the table to the OCR affiliation text and computing a similarity score. The scoring algorithm must take into account the possibility of errors in the OCR affiliation such as character substitutions, omissions or inclusions, the possibility of text in the OCR affiliation that is irrelevant to the final affiliation, such as "Dr. Smith is from...", and the possibility that the OCR text includes affiliations for authors other than the first author.

A partial-matching algorithm was developed that uses an edit distance threshold on a word by word basis, and then finds chains of such partially-matched words. The similarity score is calculated as the ratio of the total number of words in the two longest chains of partially-matched words to the number of words in the shorter of the two affiliations being compared. Tweaking the

algorithm to exclude short words, a few stop words, all-digit sequences and email addresses yielded promising results when tested with a small set of ground truth data extracted from records processed by the MARS system [8]. "Good" results are when a threshold for the calculated similarity score reliably separates the affiliations from the table that are the same institution as the OCR affiliation from the affiliations from the table that are not.

The following two examples are cases where high similarity scores correctly indicate the same institutions, i.e. true positives.

Example 1:

OCR Text:

The Waitler & Eli-a Hall Ins.titfte of Medical Researchl, Post Office Box the Royal Melboulrllle Hospital 3050, Victoriast, Atustralia

Found Affiliation:

The Walter and Eliza Hall Institute of Medical Research, Post Office Box the Royal Melbourne Hospital 3050, Victoria, Australia.

Verified Affiliation:

The Walter & Eliza Hall Institute of Medical Research, Royal Melbourne Hospital, Victoria, Australia.

In example 1, the OCR text contains several errors plus some extraneous text. The found affiliation contains no incorrect text.

Example 2:

OCR Text:

' Laboratori de Quitnica Farmaceutica, Facultat de Farmacia, Universitat de Barcelona, Avda Diagonal s/n, E-08028 Barcelona, Spain
b Laboratoire de Chinlie Generale, Consertatoire National des Arts et Mtiers, 292, rue Saint-Martin, F-75141 Paris, France

Found Affiliation:

Laboratori de Qu`imica Farmac`eutica, Facultat de Farm`acia, Universitat de Barcelona, Spain.

Verified Affiliation:

Laboratori de Qu`imica Farmac`eutica, Facultat de Farm`acia, Universitat de Barcelona, Spain.

The OCR text in example 2 contains many errors, including missing diacritics, and an extra affiliation. The found affiliation is correct as is, including the diacritical marks.

There are also cases where high similarity scores result from comparing affiliations that are not the same. The following two examples are cases where high similarity scores are associated with two different institutions, i.e. false positives.

Example 3:
OCR Text:
*Departments of Pneumonology and t Oncology and
Radiotherapy, Medical University of Gdansk, Poland

Found Affiliation:
Department of Oncology and Radiotherapy,
Medical University of Gdansk, Poland.

Verified Affiliation:
Department of Pneumonology,
Medical University of Gdansk, Poland.

In example 3, the first of the two departments listed in the OCR text is the correct affiliation, but the second department results in a higher score because those words are adjacent to the rest of the string.

Example 4:
OCR Text:
Bone Marroiw and Stem Cell Transplantation Center,
Emory University, Atlanta. GA. USA

Found Affiliation:
Denver, CO 80262, USA.
boyer_e@hub.tch.harvard.edu

Verified Affiliation:
Bone Marrow and Stem Cell Transplantation Center,
Emory University, Atlanta, GA, USA.

Example 4 illustrates the downside of excluding certain words from the partial-matching algorithm. After removing short words, email addresses and all-digit sequences from the found affiliation, the only word left is “Denver”, which is an edit distance of two from “Center”. Because the denominator of the final calculation is the shorter string, the similarity score is: $1 \text{ (word in the longest matching string of words)} / 1 \text{ (word in the shorter affiliation)} = 1.0$, the highest possible score.

There are also cases where low similarity scores result from comparing affiliations that are the same institution. The following two examples are cases where low similarity scores are calculated for the same institutions, i.e. false negatives.

Example 5:
OCR Text:
Lehrstuhl flir Titerzucht und Allen,leiler
LandwirtschlaA

Found Affiliation:
Lehrstuhl fur Tierzucht und Allgemeine
Landwirtschaftslethe, Universit“at M“unchen, Germany.
Detlef.Pietrowski@gen.vet-med.uni-muenchen.de

Verified Affiliation:
Lehrstuhl f“ur Tierzucht und Allgemeine
Landwirtschaftslethe, Universit“at M“unchen, Germany.

The OCR text of example 5 is so poor that only one of the longer words, “Lehrstuhl”, is within an edit distance of 3 (the edit distance threshold in effect for these tests) of one of the words in the found affiliation, “Lehrstuhl”. The similarity score is $1 \text{ (word in the longest matching string of words)} / 5 \text{ (words in the shorter affiliation)} = 0.2$.

Example 6:
OCR Text:
Zur ch Universty Psych atric Hospital, Zurich
Swizerliand

Found Affiliation:
Zurich University Psychiatric Hospital, Zurich,
Switzerland.

Verified Affiliation:
Zurich University Psychiatric Hospital, Zurich,
Switzerland.

The OCR server occasionally inserts extra spaces when converting italic text. Although the individual characters in the OCR of example 6 are mostly correct, two significant words are split by extraneous spaces, resulting in a similarity score of $3 \text{ (words in the longest matching string of words)} / 6 \text{ (words in the shorter affiliation)} = 0.5$.

3 Current Work

Our current efforts are focused on improving the similarity scoring module to correctly identify more of the same institutions (reduce the number of false negatives) and correctly exclude more of the institutions that are in fact different (reduce the number of false positives).

Toward that end, our first task was to create a larger set of ground truth data to use for testing. Data records were copied from thirty-one journals that had been processed by the MARS system. Some of these are known to have poor quality OCR of the affiliation field in one or more articles. Of the 650 total articles in these journals, the author name for 436 articles was found in the author/affiliation table. A dataset was constructed to include the OCR text, the found affiliation text and the

verified text for each of the found affiliations for this set of 436 articles. The dataset of 2310 records includes one or more records per article depending on how many affiliations are associated with the author name.

A special program was written to display each found affiliation text and corresponding verified affiliation text from the dataset and record a human operator's decision of whether or not they are the same institution. The decisions were accumulated for five separate operators. For each pair, if the majority of the operators determined that they are the same institution, a '1' was appended to the record in the dataset. Otherwise, a '0' was appended to the record. The dataset now became a ground truth set suitable for use in testing modifications to the similarity scoring module. 514 (22.3%) of the OCR affiliation/found affiliation pairs in the test set are the same institution. The other 1796 pairs are not.

Our first exploration of the similarity scoring module was to revisit the "bag of words" method to matching. In this approach, rather than look for strings of partially-matched words, we only look for a partially-matched occurrence in the OCR affiliation text of each significant word in the found affiliation text. The similarity score is the number of such words divided by the number of significant words in the found affiliation. "Significant" words are those that satisfy the minimum word length requirement, are not in the stop word list, are not enclosed in parentheses and are not an email address. To reduce the possibility of a strong match between the words in the found affiliation and words in the OCR affiliation text that are from affiliations other than the first affiliation, the OCR affiliation text is further edited if it is longer than 1.5 times the length of the found affiliation: ending lines of the OCR affiliation are successively removed until the remaining text is no longer than 1.5 times the length of the found affiliation. As expected, the bag of words method does calculate a low similarity score for many non-same institutions. However, higher scores do not conclusively indicate that the text being compared is for the same institution.

The bag of words method was tested with four minimum word length and minimum edit distance requirements. The resulting true positives, false positives, true negatives and false negatives for a similarity score threshold of 0.85 are shown in Figure 3. The threshold is selected by plotting all of the test results and visually choosing the score at which there is a sharp increase in the number of true positives and a sharp decrease in the number of true negatives.

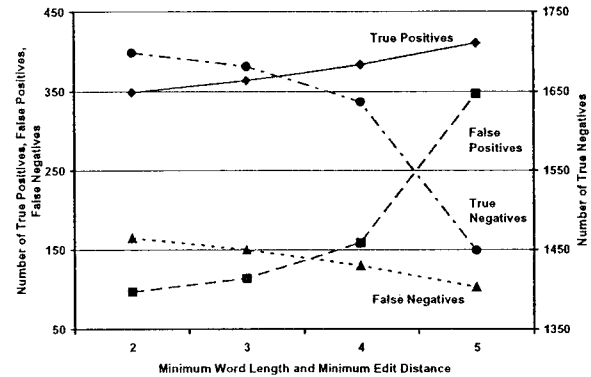


Figure 3. The effect of minimum word length and minimum edit distance on true and false positives and negatives using *bag of words* partial matching with a similarity score threshold of 0.85.

We also reexamined the original algorithm that calculated the similarity score as the ratio of the total number of words in the two longest chains of partially-matched words to the number of words in the shorter of the two affiliations being compared. This time we truncated long OCR affiliations as described for bag of words matching, and we did not remove all-digit words from consideration, reasoning that zip codes and street addresses could be useful components of a chain of partially matched words.

The chains of words method was tested with four minimum word length and minimum edit distance requirements. The resulting true positives, false positives, true negatives and false negatives for a similarity score threshold of 0.80 are shown in Figure 4. The method for selecting the threshold is the same as for Figure 3.

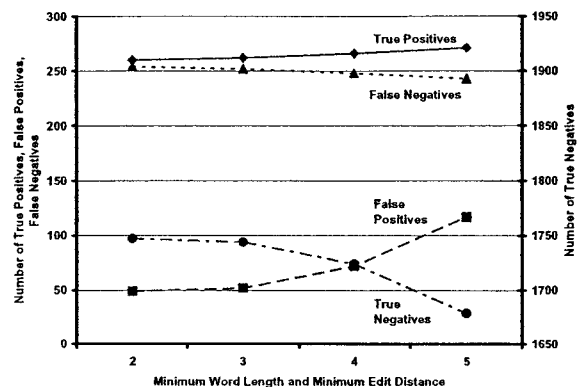


Figure 4. The effect of minimum word length and minimum edit distance on true and false positives and negatives using *chains of words* partial matching with a similarity score of 0.80.

The true positive and false positive values from Figures 3 and 4 are shown together in Figure 5.

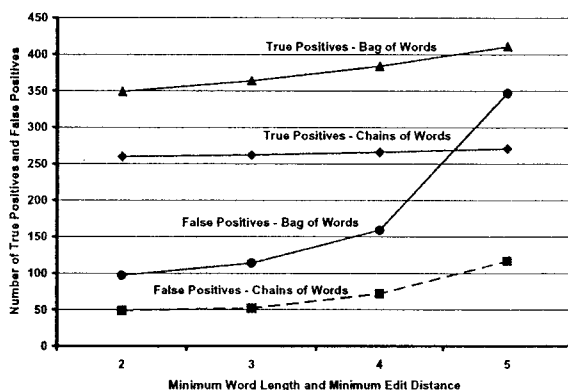


Figure 5. Comparing *bag of words* and *chains of words* method.

For both matching methods, true and false positives increase as minimum word length and minimum edit distance increase. However, true positives increase approximately linearly as a function of these requirements, while false positives increase as a power function. The tradeoff between true and false positives even extends to the choice of method: the bag of words method achieves higher rates of true positives at the cost of higher rates of false positives. However, at minimum word length and edit distance of less than 5, the ability of each method to separate the true and false positives is about the same, i.e., the number of true positives minus the number of false positives is about the same.

4 Future Strategies

There are several ideas to explore that have the potential for improving the similarity scoring algorithm. These include:

- Try other combinations of minimum word length and minimum edit distance, for example, a minimum word length of 3 and a minimum edit distance of 2.
- Explore a two-pass scheme in which the bag of words method, with one set of minimum requirements, is used to eliminate obvious non-matching affiliations, and then use the chains of words method, with another set of minimum requirements, to score the rest.
- Weight the contribution to the final score based on the word that is partially matched. For example, the

words “Department” and “University” are less useful in identifying a particular institution than the words “Immunology” and “California.” We have frequencies of word occurrence in MEDLINE affiliation fields [5] that can be used to build a table of weights. Additional information may be obtained through syntactical analysis of the found affiliation for words that are important for identifying that institution.

5 Conclusions

Simple partial matching schemes are adequate for eliminating obvious mismatches between the affiliation in the OCR text and affiliations found in the historical data. Likewise, they are adequate for finding matches in cases where the quality of the OCR text is good enough for partial matching of most of the significant words. Other techniques will be needed to reliably match affiliations for those cases where the OCR text is poor, or includes multiple affiliations. We continue to explore such techniques with confidence that we will be able to achieve reliable selection of affiliations to reduce the manual labor of the verification operator.

References

- [1] Thoma GR. Automating data entry for an online biomedical database: a document image analysis application, *Proc. 5th International Conference on Document Analysis and Recognition (ICDAR'99)* (Bangalore, India, 1999) 370-3.
- [2] Thoma GR. Automating the production of bibliographic records for MEDLINE, an R&D report of the Communications Engineering Branch, LHCBC, NLM, Bethesda, Maryland (2001) 91 pp. (archive.nlm.nih.gov)
- [3] Kim J, Le DX, Thoma GR. Automated labeling of bibliographic data extracted from biomedical online journals, *Proc. SPIE: Document Recognition and Retrieval X 5010* (January 2003) 47-56.
- [4] Thoma GR, Ford G. Automated data entry system: performance issues, *Proc. SPIE: Document Recognition and Retrieval IX 4670* (January 2002) 181-90.
- [5] Ford G, Hauser SE, Le DX, Thoma GR. Pattern matching techniques for correcting low confidence OCR words in a known context, *Proc. SPIE: Document Recognition and Retrieval VIII 4307* (January 2001) 241-9.

- [6] Lasko TA, Hauser SE. Approximate string matching algorithms for limited-vocabulary OCR output correction, *Proc. SPIE: Document Recognition and Retrieval VIII* **4307** (January 2001) 232-40.
- [7] Schlaifer J, Hauser SE. OCR affiliations: Feasibility considerations and numerical scoring for correction from past datasets. Internal project report of the Communications Engineering Branch, LHCNBC, NLM, Bethesda, Maryland (2001).
- [8] Hauser SE, Schlaifer J, Sabir TF, Demner-Fushman D, Thoma GR. Correcting OCR text by association with historic datasets, *Proc. SPIE: Document Recognition and Retrieval X* **5010** (January 2003) 84-93.

Document Analysis Resources

Balanced Query Methods for Improving OCR-Based Retrieval

Kareem Darwish

Electrical and Computer Engineering Dept.
University of Maryland, College Park
College Park, MD 20742
kareem@glue.umd.edu

Douglas W. Oard

College of Information Studies and UMIACS
University of Maryland, College Park
College Park, MD 20742
oard@glue.umd.edu

Abstract

Since many documents are available only print, improving OCR-based retrieval of scanned documents is an important problem. This paper presents a novel language independent technique for mapping queries from an error-free space to an OCR-degraded document space using a noisy channel model to produce possible degraded versions of query terms. The new technique yielded statistically significant improvements in retrieval effectiveness of as much as 39% over clean queries when tested on an Arabic document image collection.

1 Introduction

Information Retrieval (IR) is the process of satisfying a searcher's information need, expressed in the form of a query, by identifying documents that are likely to contain desired information. In many IR applications, queries and documents are represented in feature spaces that are related, but not identical. Well known examples include cross-language information retrieval (the process of finding documents in one language based on queries in a different language [4]), speech-based retrieval, and OCR-based retrieval. In such applications, a noisy channel model can be used to map between the query and document spaces, either mapping the queries into the document space or vice versa.

Much work has been done on correcting OCR errors in documents, which amounts to mapping the documents into the query space, in order to improve retrieval effectiveness (c.f., [7]). There has, however, been far less work on mapping queries into the document space based on the ways OCR would be likely to have misrecognized a term. This paper presents a novel technique for mapping queries from an error-free query space to a OCR-degraded document space using a noisy channel model for OCR degradation. The model is used to produce possible degraded versions of query terms to replace the original terms. The new technique is tested on an Arabic

document image collection at three degradation levels.

2 Previous Work

All of the work that we are aware of on the problem of mapping query terms to OCR-degraded representations has adopted an approach based on term clustering. Harding et al. computed the q-gram distance for all the terms in an OCR-degraded collection to each of the query terms. Collection terms that we found to be "near" a query term were then treated as synonyms of that term when forming the query [2]. They used the InQuery synonym operator for this purpose, which separately computes the term frequency and the document frequency of a query term as if all of the synonyms were identical. With this approach, Harding et al. reported statistically significant improvement in mean average precision (a commonly reported retrieval effectiveness measure) of 12% over use of only the query term for English words with four relatively small test collections.

A similar approach was used by Hawking, again for English, in which each query term was mapped into all the terms in the degraded documents within some fixed edit distance [3]. The edit distance measure was constrained somewhat, with the set of possible substitutions constrained to a manually constructed set of the most likely character confusions. For example, the letter "o" could be substituted for "e," but not for "l." This approach resulted in a statistically significant improvement in retrieval effectiveness over the use of only the uncorrupted query terms on a relatively large collection (from TREC-4), again using English words.

3 Methodology

The experiments reported in this paper were conducted on a relatively small collection of Arabic text that we call "Zad," for which we have both scanned pages and electronic text as ground truth [1]. The collection is comprised of 2,730 documents extracted from *Zad Al-Me'ad*, a printed book for which an accurately character coded electronic version (the "clean text") is also available [5]. Three sets of OCR outputs for the same

documents were available: print resolution (300x300 dots per inch (dpi)) as originally scanned, and down-sampled versions that approximated fine fax resolution (200x200 dpi) and standard fax resolution (200x100 dpi). The test collection includes 25 written topic descriptions and associated relevance judgments. All diacritics (short vowels) were removed and character normalizations were performed as described by Darwish and Oard [1]. Three sets of experiments were run with different index terms: character 3-grams (3g), character 4-grams (4g), and lightly stemmed words (1s). Darwish and Oard found those index terms to be among the most effective for OCR-based retrieval of Arabic [1]. All the experiments were performed using PSE, a freely redistributable vector space system designed for experimental evaluation of information retrieval algorithms. PSE uses the well-known Okapi BM-25 formula to compute term weights [6].

OCR degradation was modeled with a position-sensitive unigram character distortion model trained on a sample of real OCR results and the corresponding clean text. Automatic alignment between the OCR-degraded text and the associated clean text was used to model a manual correction process, as might be used to learn character distortion models in real applications. Experiments were run with aligned sets of between 500 and 20,000 words in order to characterize the sensitivity of the techniques being evaluated to the amount of available training data. The appearance of Arabic characters varies with position, so distortion probabilities were separately modeled for beginning, middle, end, isolated characters.

Formally, given a clean word with characters $C_1..C_i..C_n$ and the resulting word after OCR degradation $D_1..D_j..D_m$, where D_j resulted from C_i , ϵ represents the null character, and L is the position of the letter in the word (beginning, middle, end, or isolated), the three edit operations for the models would be:

$$P_{\text{substitution}}(C_i \rightarrow D_j | L) = \frac{\text{count}(C_i \rightarrow D_j | L)}{\text{count}(C_i | L)}$$

$$P_{\text{deletion}}(C_i \rightarrow \epsilon | L) = \frac{\text{count}(C_i \rightarrow \epsilon | L)}{\text{count}(C_i | L)}$$

$$P_{\text{insertion}}(\epsilon \rightarrow D_j | L) = \frac{\text{count}(\epsilon \rightarrow D_j | L)}{\text{count}(C_i | L)}$$

If the count in the numerator was zero, the computation was repeated without conditioning on position.

A separate model was trained for each resolution (print, fine fax, and standard fax). Two factors made automatic alignment of the OCR output to the clean text challenging. First, the printed and clean text versions in the Zad collection were obtained from different sources that exhibited minor differences (mostly substitution or deletion of particles such as *in*, *from*, *or*, and *then*).

Second, some areas in the scanned images of the printed page exhibited image distortions that resulted in relatively long runs of OCR errors. The alignment was performed using SCLITE from the National Institute of Standards and Technology (NIST). SCLITE employs a dynamic programming string alignment algorithm that attempts to minimize the Levenshtein distance (edit distance) between two strings. Conceptually, the algorithm uses identical matches to anchor alignment, and then uses word position with respect to those anchors to estimate an optimal alignment on the remainder of the words.

SCLITE was originally developed for speech recognition applications, but in OCR applications additional character-level evidence is available. SCLITE alignments were therefore accepted only if the number of character edit operations was less than or equal to 50% of the length of the shorter of the two matched words. To align the words that were not aligned by SCLITE the following algorithm was used:

1. Using the existing alignments as anchors, given an unaligned word at position l from the preceding anchor in a clean document, sequentially compare it to the words, in the corresponding degraded document between the corresponding pair of anchors with position l' from the preceding anchor where $|l'-l| \leq 5$.
2. When comparing two words, if the difference between their respective word lengths was less than or equal to 2 characters and the number of edit operations between the two words (using Levenshtein's edit distance) was less than a certain percentage q of the word length of the shorter one (the percentage q was the number of edit operation divided by the length of the shorter word), then the newly aligned words were used as anchors. Initially, q was set to 60%.
3. Steps 1 and 2 were iterated two more times using the new anchors with q equal to 40% and 20% to attempt to find more alignments.

This alignment technique works well for the print and fine fax resolutions, but it is a significant source of errors for highly degraded cases (e.g., standard fax resolution).

Given a pair of aligned words, they were aligned at the character level by finding the edit distance between them using the Levenshtein edit distance algorithm and then back-tracing the algorithm to identify insertions, deletions, and substitutions.

Based on the training data, a garbler was built to read in a clean word $C_1..C_i..C_n$ and synthesize OCR degradation to produce n degraded versions of each query term $D'_1..D'_j..D'_m$, where n was varied between 1 and 50. The garbler was designed to produce "balanced queries" in which is more probable degraded versions would appear more often, and would thus have a greater

effect retrieval. For each garbled word, given a character C_i , the garbler chooses a random edit operation to perform (using a randomly seeded random number generator) based on the probability distribution for the possible edit operations for one of the models. If the chosen edit operation is insertion, the garbler picks a character to be inserted depending on the distribution of possible insertions. If the edit operation is a substitution, the model substitutes the character for another one based on the probability distributions of the possible substitutions.

The experiments were designed to answer the following questions:

1. Does mapping queries into document space improve retrieval effectiveness?
2. If so, how many training words are necessary to train the OCR-degradation model?
3. What is the optimal value of n (number of degraded versions for a query term)?

4 Results and Discussion

All the results were compared to a baseline generated using the query terms without any garbling. Baseline mean average precision results for the Zad collection are presented in Table 1. Tables 2, 3, and 4 at the end of the paper show the results for each contrastive condition. As can be seen, the method produced statistically significantly better results than the baseline for all index terms (based on a paired two-tailed t -test with $p < 0.05$). Using more than 5,000 training words was found to offer little benefit for retrieval effectiveness, but smaller training sets yielded significantly worse results. This indicates that an adequate amount of training data could be hand corrected in just a few hours. Larger numbers of garbled terms (35 or 50) yielded the best results, indicating that the improved fidelity resulting from repeated sampling was helpful.

Table 1: Baseline Results for the Zad Collection.

| Index Term | Mean Avg. Precision | | | |
|------------|---------------------|-------|----------|---------|
| | Clean | Print | Fine Fax | Std Fax |
| 3g | 0.50 | 0.44 | 0.33 | 0.23 |
| 4g | 0.53 | 0.46 | 0.32 | 0.22 |
| ls | 0.52 | 0.43 | 0.24 | 0.19 |

The maximum observed relative improvements over the clean-query baseline for the same combination of indexing term and resolution were as follows:

| Resolution | Index Term | Maximum Relative Improvement |
|--------------|------------|------------------------------|
| Print | 3g | 8% |
| | 4g | 7% |
| | ls | 4% |
| Fine Fax | 3g | 13% |
| | 4g | 29% |
| | ls | 36% |
| Standard Fax | 3g | 14% |
| | 4g | 27% |
| | ls | 22% |

Smoothing occurrence frequencies is often helpful when estimating distortion probabilities, so a second set of experiments was also tried with a simple variant of regression towards the most likely condition – in this case, the unchanged query. Document scores from the baseline runs for 3-grams, 4-grams, and lightly stemmed words were therefore combined with the document scores from the random garbling runs. When combining the scores, new document scores were:

$$\text{newScore} = \alpha * \text{baselineScore} + (1 - \alpha) * \text{garbledScore}$$

Note that the smoothing here is applied to the effect (the document score) rather than directly to the probability distribution; this is a common approach to combining evidence in information retrieval applications. To find optimal values for α , baseline runs were combined with all previous garbled runs at values of α that varied between 0.1 and 0.9 with increments of 0.1. For each condition, the value of α that produced the highest mean average precision was noted. The average of the optimal value of α was computed separately for 10, 20, 35, and 50 garbled versions of each query term, and the resulting values were then averaged to provide a single value for use with each combination of indexing term and resolution. The resulting averages values of α were as follows:

| Index Term | Print | Fine Fax | Standard Fax |
|------------|-------|----------|--------------|
| 3g | 0.4 | 0.3 | 0.4 |
| 4g | 0.2 | 0.2 | 0.5 |
| Ls | 0.5 | 0.4 | 0.5 |

The results for these combinations are summarized in Tables 5, 6, and 7. The maximum observed percent improvements over the clean-query baseline were for the same combination of indexing term and resolution were as follows:

| Resolution | Index Term | Relative Improvement |
|------------|------------|----------------------|
| Print | 3g | 7% |
| | 4g | 7% |
| | ls | 4% |
| Fine Fax | 3g | 16% |

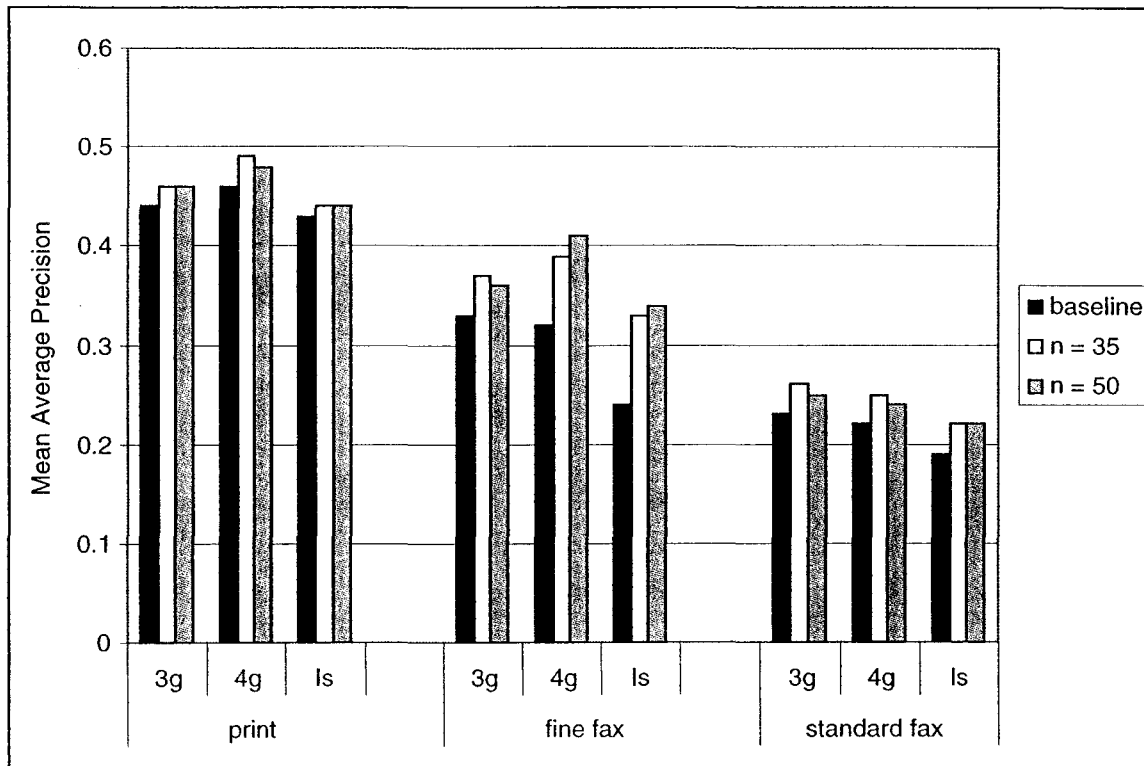


Figure 1: Combination of evidence results for different index terms using 5,000 training words

| | | |
|--------------|----|-----|
| Standard Fax | 4g | 28% |
| | ls | 39% |
| | 3g | 16% |
| | 4g | 16% |
| | ls | 25% |

Because the parameters were trained on the test collection, these should be thought of as upper bounds rather than as estimates of what can be accomplished in a real application. Little effect on the maximum relative improvement was observed, but a comparison of Tables 2-7 reveals that statistical significance was more often observed in the observed differences. Again, a greater number of garbled terms (35 or 50) yielded the best results, and providing more than 5,000 words of training data was not generally helpful.

Figure 1 shows combination of evidence results for different index terms and resolution with 5,000 training words. For print resolution, the maximum gains are modest, in part because the baseline results were excellent. Indeed, the mean average precision for the OCR-degraded collection using the mapping technique was statistically indistinguishable from retrieval of uncorrupted documents for 3-grams and 4-grams.

Balanced queries based on garbling showed the greatest relative improvements for fine fax resolution. Longer index terms (4-grams and lightly stemmed words) seemed to benefit the most, with a 39% improvement in retrieval effectiveness observed for lightly stemmed

words.

For the standard fax resolution, the balanced query garbling technique often statistically outperformed the baseline. The observed relative improvements in retrieval effectiveness were smaller than those observed for fine fax resolution, perhaps because of some incorrect alignments in the training data (a factor that would not be present with hand-corrected training data).

4 Conclusions and Future Work

This paper presented a technique for mapping query terms into an OCR-degraded document space using a noisy channel model. The technique proved to be effective across a range of degradation levels, with relative improvements as high as 39% over a baseline in which no changes were made to the clean query. The technique has been demonstrated on a relatively small Arabic test collection, but the key ideas behind the technique are language independent. At present, the only large test collections for evaluation of retrieval effectiveness in OCR-degraded collections are based on synthetic (modeled) distortion of clean test collections, so a fair evaluation on a large collection will need to await the creation of a large collections based on actual OCR. The size and statistical significance of the observed improvements on the Zad collection do, however, suggest that benefits from this technique are likely to be seen on collections of any size.

There are two obvious ways in which the work reported in this paper could be extended. First, it would be worthwhile to explore the use of InQuery's weighted sum and synonym operators. Query term weighting can achieve a finer degree of granularity than the query term replication used in the experiments reported above, and the use of the synonym operator to implement "structured queries" is known to be helpful in similar applications such as cross-language retrieval. The experiments with model-based generation of plausible alternative expressions of query terms in an OCR-degraded collection reported in this paper suggest that those additional research directions would be well worth exploring.

References

- [1] Darwish, K. and D. Oard. *Term Selection for Searching Printed Arabic*. SIGIR 2002, 261-268, 2002.
- [2] Harding, S., W. Croft, and C. Weir. *Probabilistic Retrieval of OCR-degraded Text Using N-Grams*. European Conference on Digital Libraries, 1997.
- [3] Hawking, D. *Document Retrieval in OCR-Scanned Text*. Sixth Parallel Computing Workshop, Kawasaki, Japan, November 1996.
- [4] Oard, D. and B. Dorr, *A Survey of Multilingual Text Retrieval*. UMIACS, University of Maryland, College Park, 1996.
- [5] Al-Areeb Electronic Publishers, LLC. 16013 Malcolm Dr., Laurel, MD 20707, USA.
- [6] Robertson, S. and K. S. Jones. *Simple proven approaches to text retrieval*. Tech. Rep. TR356, Cambridge University Computer Laboratory, 1997.
- [7] Taghva, K., J. Borsack, and A. Condit. *An Expert System for Automatically Correcting OCR Output*. Proceedings of the SPIE - Document Recognition, pages 270--278, 1994.

Table 2: Print Resolution – All results are of Mean Average Precision. Grey cells indicate statistically worse results than the baseline and black cells indicate statistically better results than the baseline

| Index Term | No. of Training words | baseline | 1 | 5 | 10 | 20 | 35 | 50 |
|------------|-----------------------|----------|------|------|------|------|------|------|
| 3g | 500 | 0.44 | 0.31 | 0.41 | 0.45 | 0.44 | 0.42 | 0.42 |
| | 1,000 | 0.44 | 0.40 | 0.44 | 0.45 | 0.46 | 0.44 | 0.45 |
| | 2,000 | 0.44 | 0.38 | 0.44 | 0.46 | 0.45 | 0.45 | 0.44 |
| | 5,000 | 0.44 | 0.32 | 0.47 | 0.47 | 0.46 | 0.47 | 0.44 |
| | 10,000 | 0.44 | 0.34 | 0.46 | 0.45 | 0.47 | 0.45 | 0.45 |
| | 20,000 | 0.44 | 0.39 | 0.46 | 0.48 | 0.46 | 0.46 | 0.45 |
| 4g | 500 | 0.46 | 0.29 | 0.42 | 0.46 | 0.47 | 0.47 | 0.46 |
| | 1,000 | 0.46 | 0.40 | 0.45 | 0.47 | 0.47 | 0.48 | 0.48 |
| | 2,000 | 0.46 | 0.37 | 0.47 | 0.49 | 0.48 | 0.48 | 0.47 |
| | 5,000 | 0.46 | 0.30 | 0.48 | 0.48 | 0.49 | 0.48 | 0.48 |
| | 10,000 | 0.46 | 0.32 | 0.48 | 0.48 | 0.49 | 0.49 | 0.48 |
| | 20,000 | 0.46 | 0.30 | 0.48 | 0.48 | 0.48 | 0.48 | 0.49 |
| ls | 500 | 0.43 | 0.27 | 0.39 | 0.41 | 0.43 | 0.41 | 0.42 |
| | 1,000 | 0.43 | 0.28 | 0.40 | 0.42 | 0.43 | 0.43 | 0.43 |
| | 2,000 | 0.43 | 0.25 | 0.41 | 0.44 | 0.43 | 0.43 | 0.42 |
| | 5,000 | 0.43 | 0.21 | 0.41 | 0.43 | 0.43 | 0.43 | 0.44 |
| | 10,000 | 0.43 | 0.28 | 0.38 | 0.41 | 0.43 | 0.43 | 0.43 |
| | 20,000 | 0.43 | 0.26 | 0.44 | 0.43 | 0.45 | 0.44 | 0.43 |

Table 3: Fine Fax Resolution – All results are of Mean Average Precision. Grey cells indicate statistically worse results than the baseline and black cells indicate statistically better results than the baseline

| Index Term | No. of Training words | baseline | 1 | 5 | 10 | 20 | 35 | 50 |
|------------|-----------------------|----------|------|------|------|------|------|------|
| 3g | 500 | 0.33 | 0.19 | 0.31 | 0.32 | 0.34 | 0.35 | 0.34 |
| | 1,000 | 0.33 | 0.20 | 0.31 | 0.34 | 0.35 | 0.37 | 0.36 |
| | 2,000 | 0.33 | 0.15 | 0.32 | 0.35 | 0.36 | 0.35 | 0.35 |
| | 5,000 | 0.33 | 0.14 | 0.29 | 0.32 | 0.37 | 0.36 | 0.35 |
| | 10,000 | 0.33 | 0.16 | 0.29 | 0.33 | 0.35 | 0.35 | 0.36 |
| | 20,000 | 0.33 | 0.21 | 0.30 | 0.34 | 0.35 | 0.36 | 0.36 |
| 4g | 500 | 0.32 | 0.22 | 0.34 | 0.36 | 0.36 | 0.38 | 0.39 |
| | 1,000 | 0.32 | 0.21 | 0.33 | 0.35 | 0.37 | 0.40 | 0.39 |
| | 2,000 | 0.32 | 0.22 | 0.32 | 0.37 | 0.37 | 0.40 | 0.39 |
| | 5,000 | 0.32 | 0.22 | 0.29 | 0.36 | 0.39 | 0.40 | 0.41 |
| | 10,000 | 0.32 | 0.22 | 0.29 | 0.35 | 0.39 | 0.40 | 0.41 |
| | 20,000 | 0.32 | 0.23 | 0.31 | 0.35 | 0.37 | 0.41 | 0.41 |
| ls | 500 | 0.24 | 0.16 | 0.25 | 0.25 | 0.30 | 0.30 | 0.32 |
| | 1,000 | 0.24 | 0.17 | 0.23 | 0.26 | 0.29 | 0.33 | 0.32 |
| | 2,000 | 0.24 | 0.19 | 0.24 | 0.28 | 0.31 | 0.31 | 0.30 |
| | 5,000 | 0.24 | 0.17 | 0.23 | 0.26 | 0.31 | 0.30 | 0.32 |
| | 10,000 | 0.24 | 0.17 | 0.24 | 0.24 | 0.30 | 0.31 | 0.31 |
| | 20,000 | 0.24 | 0.17 | 0.23 | 0.24 | 0.28 | 0.30 | 0.32 |

Table 4: Standard Fax Resolution – All results are of Mean Average Precision. Grey cells indicate statistically worse results than the baseline and black cell indicate statistically better results than the baseline

| Index Term | No. of Training words | baseline | 1 | 5 | 10 | 20 | 35 | 50 |
|------------|-----------------------|----------|------|------|------|------|------|------|
| 3g | 500 | 0.23 | 0.19 | 0.19 | 0.22 | 0.23 | 0.24 | 0.23 |
| | 1,000 | 0.23 | 0.19 | 0.22 | 0.20 | 0.21 | 0.24 | 0.24 |
| | 2,000 | 0.23 | 0.19 | 0.20 | 0.22 | 0.24 | 0.23 | 0.23 |
| | 5,000 | 0.23 | 0.19 | 0.21 | 0.24 | 0.26 | 0.22 | 0.22 |
| | 10,000 | 0.23 | 0.22 | 0.22 | 0.25 | 0.24 | 0.24 | 0.24 |
| | 20,000 | 0.23 | 0.19 | 0.19 | 0.22 | 0.24 | 0.24 | 0.24 |
| 4g | 500 | 0.22 | 0.19 | 0.19 | 0.20 | 0.25 | 0.23 | 0.25 |
| | 1,000 | 0.22 | 0.19 | 0.21 | 0.22 | 0.23 | 0.25 | 0.26 |
| | 2,000 | 0.22 | 0.17 | 0.20 | 0.23 | 0.25 | 0.25 | 0.25 |
| | 5,000 | 0.22 | 0.21 | 0.23 | 0.24 | 0.27 | 0.25 | 0.25 |
| | 10,000 | 0.22 | 0.21 | 0.22 | 0.23 | 0.25 | 0.27 | 0.27 |
| | 20,000 | 0.22 | 0.19 | 0.19 | 0.23 | 0.25 | 0.28 | 0.28 |
| ls | 500 | 0.19 | 0.16 | 0.16 | 0.17 | 0.18 | 0.20 | 0.22 |
| | 1,000 | 0.19 | 0.16 | 0.17 | 0.17 | 0.19 | 0.23 | 0.23 |
| | 2,000 | 0.19 | 0.14 | 0.15 | 0.18 | 0.21 | 0.21 | 0.21 |
| | 5,000 | 0.19 | 0.17 | 0.18 | 0.19 | 0.22 | 0.22 | 0.21 |
| | 10,000 | 0.19 | 0.16 | 0.17 | 0.19 | 0.20 | 0.22 | 0.22 |
| | 20,000 | 0.19 | 0.15 | 0.15 | 0.18 | 0.21 | 0.21 | 0.21 |

Table 5: Print Resolution -- Combination of Evidence – All results are of Mean Average Precision. Black cell indicate statistically better results than the baseline

| Index Term | No. of Training words | baseline | 10 | 20 | 35 | 50 |
|------------|-----------------------|----------|------|------|------|------|
| 3g | 500 | 0.44 | 0.45 | 0.45 | 0.44 | 0.44 |
| | 1,000 | 0.44 | 0.44 | 0.46 | 0.45 | 0.46 |
| | 2,000 | 0.44 | 0.46 | 0.46 | 0.45 | 0.45 |
| | 5,000 | 0.44 | 0.46 | 0.46 | 0.46 | 0.46 |
| | 10,000 | 0.44 | 0.46 | 0.47 | 0.46 | 0.45 |
| | 20,000 | 0.44 | 0.47 | 0.47 | 0.47 | 0.47 |
| 4g | 500 | 0.46 | 0.46 | 0.47 | 0.48 | 0.47 |
| | 1,000 | 0.46 | 0.47 | 0.47 | 0.48 | 0.49 |
| | 2,000 | 0.46 | 0.49 | 0.48 | 0.48 | 0.48 |
| | 5,000 | 0.46 | 0.48 | 0.49 | 0.49 | 0.48 |
| | 10,000 | 0.46 | 0.49 | 0.49 | 0.49 | 0.49 |
| | 20,000 | 0.46 | 0.49 | 0.49 | 0.49 | 0.49 |
| ls | 500 | 0.43 | 0.43 | 0.43 | 0.43 | 0.43 |
| | 1,000 | 0.43 | 0.43 | 0.44 | 0.44 | 0.44 |
| | 2,000 | 0.43 | 0.44 | 0.44 | 0.44 | 0.44 |
| | 5,000 | 0.43 | 0.44 | 0.44 | 0.44 | 0.44 |
| | 10,000 | 0.43 | 0.44 | 0.44 | 0.44 | 0.44 |
| | 20,000 | 0.43 | 0.44 | 0.44 | 0.45 | 0.44 |

Table 6: Fine Fax Resolution -- Combination of Evidence – All results are of Mean Average Precision. Black cell indicate statistically better results than the baseline

| Index Term | No. of Training words | baseline | 10 | 20 | 35 | 50 |
|------------|-----------------------|----------|------|------|------|------|
| 3g | 500 | 0.33 | 0.33 | 0.34 | 0.35 | 0.36 |
| | 1,000 | 0.33 | 0.34 | 0.36 | 0.37 | 0.37 |
| | 2,000 | 0.33 | 0.36 | 0.37 | 0.36 | 0.38 |
| | 5,000 | 0.33 | 0.34 | 0.37 | 0.37 | 0.36 |
| | 10,000 | 0.33 | 0.35 | 0.37 | 0.37 | 0.39 |
| | 20,000 | 0.33 | 0.35 | 0.36 | 0.37 | 0.37 |
| 4g | 500 | 0.32 | 0.35 | 0.36 | 0.36 | 0.39 |
| | 1,000 | 0.32 | 0.34 | 0.36 | 0.39 | 0.39 |
| | 2,000 | 0.32 | 0.37 | 0.38 | 0.40 | 0.39 |
| | 5,000 | 0.32 | 0.36 | 0.38 | 0.39 | 0.41 |
| | 10,000 | 0.32 | 0.38 | 0.38 | 0.40 | 0.40 |
| | 20,000 | 0.32 | 0.37 | 0.38 | 0.40 | 0.41 |
| ls | 500 | 0.24 | 0.26 | 0.31 | 0.29 | 0.33 |
| | 1,000 | 0.24 | 0.29 | 0.30 | 0.32 | 0.33 |
| | 2,000 | 0.24 | 0.30 | 0.32 | 0.32 | 0.32 |
| | 5,000 | 0.24 | 0.30 | 0.32 | 0.33 | 0.34 |
| | 10,000 | 0.24 | 0.28 | 0.31 | 0.32 | 0.33 |
| | 20,000 | 0.24 | 0.28 | 0.30 | 0.33 | 0.34 |

Table 7: Standard Fax Resolution -- Combination of Evidence – All results are of Mean Average Precision. Black cell indicate statistically better results than the baseline

| Index Term | No. of Training words | baseline | 10 | 20 | 35 | 50 |
|------------|-----------------------|----------|------|------|------|------|
| 3g | 500 | 0.23 | 0.24 | 0.24 | 0.25 | 0.26 |
| | 1,000 | 0.23 | 0.24 | 0.23 | 0.25 | 0.24 |
| | 2,000 | 0.23 | 0.23 | 0.24 | 0.25 | 0.25 |
| | 5,000 | 0.23 | 0.23 | 0.24 | 0.26 | 0.25 |
| | 10,000 | 0.23 | 0.26 | 0.25 | 0.25 | 0.26 |
| | 20,000 | 0.23 | 0.23 | 0.25 | 0.26 | 0.26 |
| 4g | 500 | 0.22 | 0.22 | 0.23 | 0.24 | 0.24 |
| | 1,000 | 0.22 | 0.23 | 0.23 | 0.24 | 0.25 |
| | 2,000 | 0.22 | 0.22 | 0.23 | 0.24 | 0.24 |
| | 5,000 | 0.22 | 0.23 | 0.24 | 0.25 | 0.24 |
| | 10,000 | 0.22 | 0.24 | 0.23 | 0.24 | 0.25 |
| | 20,000 | 0.22 | 0.22 | 0.23 | 0.25 | 0.25 |
| ls | 500 | 0.19 | 0.20 | 0.20 | 0.21 | 0.22 |
| | 1,000 | 0.19 | 0.21 | 0.21 | 0.23 | 0.23 |
| | 2,000 | 0.19 | 0.20 | 0.21 | 0.22 | 0.22 |
| | 5,000 | 0.19 | 0.21 | 0.21 | 0.22 | 0.22 |
| | 10,000 | 0.19 | 0.20 | 0.20 | 0.21 | 0.23 |
| | 20,000 | 0.19 | 0.19 | 0.20 | 0.23 | 0.22 |

Creation of Multi-Lingual data resources and evaluation tool for OCR

S. Setlur S. Kompalli R. Vemulapati V. Govindaraju

Center of Excellence for Document Analysis and Recognition,

University at Buffalo, Buffalo, NY - 14260

{kompalli, setlur, govind, raman-v}@cedar.buffalo.edu

Abstract

Indic scripts contribute to a large number of languages and dialects used by people all over the world with Devanagari being most widely used. To date, OCR techniques of different Indic scripts have not yet been widely disseminated or evaluated independently. Automated evaluation tools are currently not available for lack of a standard representation of ground-truth and result data.

A key reason for the absence of sustained research efforts in off-line Devanagari OCR appears to be the paucity of data resources. Ground truthed data for words and characters, on-line dictionaries, corpora of text documents and reliable, standardized statistical analysis and evaluation tools are currently lacking. So, the creation of such data resources will undoubtedly provide a much needed fillip to researchers working on Devanagari OCR.

This paper describes a National Science Foundation sponsored project under the International Digital Libraries program to create data resources that will facilitate development of Devanagari OCR technology and provide a standardized test bed and evaluation tools for Devanagari script recognition.¹

1 Introduction

Automatic text recognition using OCR is the process of converting the image of a textual document into its digital textual equivalent, the advantage being that the textual material can be edited. The document image itself can be either machine-printed or handwritten, or a combination of the two.

The recognition of any unit of a document is in the context of the rest of the document and can benefit from the context. For all practical purposes, the units of recognition range from a single character

¹This material is based upon work supported by the National Science Foundation under Grant No. 0112059. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

to a complete sentence. While character images are intuitively the default domain in which OCR's operate, grammatical and linguistic constraints from the sentence level can drive the recognition of the characters and words [1, 12].

The development of OCRs requires vast amounts of training data with ground-truth to achieve acceptable levels of performance. Manual truthing of data is expensive and labor intensive. Moreover, a standardized, public-domain truth set is essential to to justly compare the relative performance of different OCRs. Such data is sorely lacking for Indic Script OCR researchers at present. In this paper, we discuss the architecture of a system to collect data for *Devanagari*, the most widely used Indic Script. Section 2 provides a brief overview of the *Devanagari* script. Section 3 reviews some *Devanagari* OCR techniques that have been published. Section 4 describes the data collection and ground-truthing methodology and the tool designed for this purpose. Section 6 describes the future work planned under this initiative.

2 Overview of Devanagari script

The basic character set of *Devanagari* script is made up of 16 vowels (Table 1) and 36 basic consonants (Table 2). A vowel following a consonant takes a modified shape, which is placed to the left, right, top or bottom of the consonant. Each of the consonants combines with vowel modifiers for 15 of the 16 vowels to create new character shapes (Table 1). More than one consonant can combine with each other to form conjunct consonants (Some examples can be seen in Table 3). In addition to these, there are also a few special characters, which are shown in Table 2 and 10 numerals shown in Table 2. Thus, the total number of character shapes encountered in *Devanagari* script is well over 300 and it is evident from looking at the characters that there would be a significant confusion between similar characters. A word in *Devanagari* script consists of composite charac-

Table 1: *Devanagari* vowels and their corresponding consonant modifiers

| Vowel | Modifier | Modified Consonant | Vowel | Modifier | Modified Consonant |
|-------|----------|--------------------|-------|----------|--------------------|
| अ | | क | लृ | ३ | |
| आ | । | का | लृ | ३ | |
| इ | ि | कि | ए | १ | के |
| ई | ी | की | ऐ | १ | कै |
| उ | ु | कु | ओ | १ | को |
| ऊ | ू | कू | औ | १ | कौ |
| ऋ | ृ | कृ | अं | ॰ | कं |
| ॠ | ॠ | कृ | अः | ः | कः |

Table 2: Simple Consonants (with transliteration), Digits and Special Characters in *Devanagari*

| Simple Consonants | | | | | | | | | | |
|-------------------|-----|----|-----|-----|-----|-----|----|-----|-----|-----|
| क | ख | ग | घ | ङ | च | छ | ज | झ | ञ | |
| ka | kha | ga | gha | ṅa | cha | Cha | ja | jha | JNa | |
| ट | ठ | ड | ढ | ण | त | थ | द | ध | न | |
| Ta | Tha | Da | Dha | Na | ta | tha | da | dha | na | |
| प | फ | ब | भ | म | | | | | | |
| pa | pha | ba | bha | ma | | | | | | |
| य | र | ल | व | श | ष | स | ह | ळ | क्ष | ज्ञ |
| ya | ra | la | va | sha | Sha | sa | ha | La | xa | GYa |
| Digits | | | | | | | | | | |
| ० | १ | २ | ३ | ४ | ५ | ६ | ७ | ८ | ९ | |

| Special Characters | |
|--------------------|---------------|
| ◌̣ | Chandrabindu |
| ◌̣̣ | Avagraha |
| । | Danda |
| ॥ | Paragraph end |
| ॐ | AUM |

Table 3: Examples of conjunct consonant shapes in *Devanagari*

| Consonants | Conjunct form | Consonants | Conjunct form | Consonants | Conjunct form |
|------------|---------------|------------|---------------|------------|---------------|
| क त | क्त | द द | द्द | ल ल | ल्ल |
| क य | क्य | द ध | द्ध | श च | श्च |
| क व | क्व | द न | द्व | श न | श्न |
| घ न | घ्न | द म | द्व | श र | श्च |
| ङ क | ङ्क | द य | द्व | श ल | श्च |
| ङ ग | ङ्ग | द व | द्व | श व | श्च |
| ञ च | ञ्च | ध न | ध्न | ष ट | ष्ट |
| ट ट | ट्ट | न न | न्न | ह न | ह्न |
| त त | त्त | प त | प्त | ह य | ह्य |
| त न | त्न | प न | प्न | ह र | ह्र |
| त र | त्र | प ल | प्ल | ह ल | ह्ल |
| द ग | द्ग | म ल | म्ल | ह व | ह्व |

Table 4: Indic scripts

| Script | Official | Transliteration | Unicode | Vowels | Consonants | Vowel Signs | Spl Characters |
|------------|----------|-----------------|---------|--------|------------|-------------|----------------|
| Benagali | Yes | Yes | Yes | 11 | 32 | 11 | 25 |
| Devanagari | Yes | Yes | Yes | 16 | 37 | 14 | 27 |
| Gujarati | Yes | Yes | Yes | 13 | 34 | 13 | 8 |
| Kannada | Yes | Yes | Yes | 14 | 35 | 13 | 8 |
| Tamil | Yes | Yes | Yes | 12 | 22 | 11 | 7 |
| Telugu | Yes | Yes | Yes | 14 | 35 | 13 | 8 |
| Gurumukhi | Yes | Yes | Yes | 10 | 32 | 10 | 11 |

ters joined by a horizontal line at the top which is referred to as the header line or *shirorekha*. A single or double vertical line called *danda* is traditionally used to indicate end of phrase or end of sentence. In modern practice, inter-word spacing and use of Latin punctuation prevail. Likewise, although *Devanagari* has a native set of symbols for numerals, Arabic numerals are sometimes used in modern documents.

2.1 Similarity of other Indic Scripts to Devanagari Script

Bengali, Gujarati, and Gurumukhi (Punjabi) along with Devanagari (Hindi) are classified into a group called *Aryan* scripts and hold similar characteristics with respect to their rendering, formation of compound/conjunct characters and modifiers. Tamil, Telugu and Kannada are classified as *Dravidian* scripts and their character shapes are distinct from those of *Aryan* scripts. Though there exist some differences between the two groups of scripts, for example - fewer consonants and lack of consonant forms in Tamil, due to the structural similarity of the languages, similar technologies support several Indic Scripts (Table 4). The Multi-Lingual truthing tool has been designed taking advantage of this similarity, leading to a single technique for handling the truthing of documents in these scripts.

3 Devanagari OCR - Background

We surveyed the available *Devanagari* OCR literature to study the techniques used by researchers in the area to determine the nature of data we would be required to collect for the test bed. *Devanagari* text can have compound characters, characters with vowel modifiers and half characters. Due to the complexity of the character set, Researchers pre-process *Devanagari* characters in different ways. While some attempt to separate the vowel modifiers and the core characters [18], recognizing them separately, others attempt to recognize each compound consonant or consonant with vowel modifier as a separate class [19, 14].

Research [24] into compound and fused character segmentation of *Devanagari* script indicates an accuracy level of 85%, using a hybrid, multi-pass approach that relies on structural features. Many other techniques [11, 26] also exist that give similar results, but using different algorithms. There is a strong indication that the character-segmentation level has significant issues, and algorithms handling those issues need to be analysed and tested independently.

Researchers [18] using structural features and implementing a blackboard architecture have shown solutions that provide about 70% accuracy on *Devanagari* text containing unknown font and about 80% accuracy for text with a known font. Statistical and Structural features, starting from average word and character dimensions; which were used for word/character separation to stroke information; which lead to actual character classification, were obtained from the document image. These were taken together and further analyzed to obtain improved results; identifying first the base characters, then the compound characters etc. Thus, it was a kind of re-entrant algorithm. Other researchers [14], used a top-down architecture, to achieve about 96% accuracy at the character recognition level. Various structural features of *Devanagari* script were used to create a heuristic and obtain character stroke features. These feature sets were then processed using a Decision-tree classifier. Other techniques include a Binary tree classifier based on basic primitive constructs for the script [9], and syntactic pattern analysis systems, which used a picture language for *Devanagari* OCR [10].

Current research in *Devanagari* OCR does not indicate a single view of what the granularity of a character class should be, with some adopting the notion of treating the base character distinct from the modifier [8, 18] and others treating the base character with the modifiers or conjunct characters as distinct classes [11, 14].

Given these issues, it was decided to collect word level data and ground-truth as a start and words

would be later segmented into their component characters and ground-truthed as necessary. This is a significant departure from previous databases [3, 2, 7, 6] which mostly provided character level data only. However, the scripts listed in section 3.1 by nature, do not have compound characters and their words construction is made up of a much simpler, sequential ordering of individual characters.

3.1 Current OCR databases

Current OCR training data sets provide samples of all characters in the script under consideration. A few of the databases listed below are indicative that such an exhaustive coverage of all characters is not limited to Latin-character based scripts [3, 2], but is retained across other, complicated scripts. [7, 6]

- UW English Document Image Database I
1147 document page images from English Scientific and Technical Journals, with text zone bounds, ground truth data for each zone, finer attributes and qualitative information. [3]
- Handwritten Korean character image database
Handwritten Cursive Hangul character images - 2,350 classes * 100 samples, 100dpi grayscale/binary images. [7]
- UW-II English/Japanese Document Image Database
624 English pages from journals, 477 Japanese pages from journals, with text zone bounds, ground truth data for each zone, finer attributes and qualitative information. [4]
- ERIM Arabic Document Database
Over 750 pages of Arabic text at 300dpi, all character information stored as Unicode, Truth for characters as well as ligatures, Over 200 distinct Arabic ligatures Approximately 1,000,000 characters. [5]
- Japanese Character Image Database
Approximately 180,000 Kanji, Hiragana, Katakana, alphanumeric, and symbolic characters extracted from a variety of machine printed documents, with quality varying from clean to degraded digitized as binary tiff images at 400 dpi on a flatbed scanner. More than 3,300 character categories in JIS level-0 and level-1. Every image supplied with a manually determined truth-value (4-byte JIS code). [6]

3.2 Other data Visualization tools

This section summarizes the tools that were studied in order to ensure platform independence and ease of

user interaction. The survey included Multi-Lingual text editors, OCR's and other truth data visualization toolkits.

- iLEAP, a Indic language word processor
Developed by CDAC, India, iLEAP is an Intelligent, Internet ready, Indian language Word Processor on Windows. The desktop view of iLEAP has been made self explanatory, with language-sensitive tool layout and operation. When the choice of language is Hindi, the entire context of the system, including keyboard layouts, available fonts, and the word reference is changed to Hindi. It is possible to switch between the languages, and hence the context of operation at the click of a mouse.
- Chitrakan, a *Devanagari* OCR
A *Devanagari* OCR package developed by CDAC, India, Chitrakan provides Text and Picture segmentation for grayscale/color, .bmp/.tiff images scanned at 300 dpi and gives results in ISCI or .RTF format. In addition, rotation, color inversion and file printing, image editing tools are provided.
- Transcriber, for speech signal segmentation
Transcriber is a tool for segmenting, labeling and transcribing speech and provides management of several layers of segmentation, for different applications. It synchronizes the cursors in the text editor and the signal window, ensuring always that control remains on the signal that is being processed. Transcriptions are stored in XML format for easier automatic processing and exchange; syntactic validation of files upon their DTD. Transcriptions are encoded using the most current encodings including Unicode.
- ViPER Video Processing Evaluation Resource
ViPER is a toolkit to automate video processing evaluation tasks, aimed at developing a ground truth representation which can be maintained and manipulated by a number of different tools and applications. The system is modularized into three parts - ViPER-GT, for creating frame-level ground truth; ViPER-PE, providing performance evaluation of any automated video analysis system; and ViPER-Viz for visualization that contains programs which enable a user to compactly visualize ground truth results, and the performance evaluation results.
- GATE - General Architecture for text encoding.
GATE comprises an architecture, development environment, created by the Sheffield NLP

group, and has been used in many language processing projects, particularly for Information Extraction in many languages. GATE includes resources for common Language Engineering data structures and algorithms, including documents, corpora and various annotation types, a set of language analysis components for Information Extraction and a range of data visualisation and editing components. Input can be documents in a variety of formats including XML, RTF, email, HTML, SGML and plain text, and are converted into a single unified model of annotation.

4 Data collection and ground truthing of Devanagari documents

Character and word training data with ground truth from documents that are representative of those encountered in the real world are basic to the development of recognition systems. However, collection and truthing of data is a laborious and expensive process. Designing an evaluation test bed for recognition systems also requires creation of test sets that account for most, if not all of the variability seen in the real world [17].

4.1 Simulated Data

Simulated data has often been proposed as an alternative to real data for OCR evaluation. An analysis of Asian and Latin script separation using simulated data [22] indicated that at the level of separating languages, degradation models would work well. Also, research has been done in the direction of multilingual OCR evaluation using multiple, multilingual versions of a single document set (the bible) as groundtruth [23]. Multiple copies were obtained from a computer generated common dataset by using degradation models.

Earlier research [21] describes a Morphological degradation model that takes into consideration pixel inversion and blurring as modeling parameters. Another degradation model [20] relies on image and script features such as Resolution, Size, Scaling, Blur, Offset, Skew, Threshold, Jitter and Sensitivity as salient features suitable for generating simulated documents.

A survey of feature extraction techniques for OCR [25] indicates that OCR researchers make use of a very rich set of feature vectors, encompassing both variant features as well as invariant features like pixel density, rotation invariant Zernike Moments, Geometric Moment Invariants, Normalized projection histograms etc. Since different models operate on different image properties, and because different feature vectors are used by classifiers for different scripts, it is not likely that all simulation models

would work equally well for OCRs of all scripts. To the best knowledge of the authors, a detailed analysis of simulation models in the lines of work done for Latin or Japanese [22] has not yet been attempted for *Devanagari*. The design of the truthing tool has been such that various degradation parameters for the documents can be measured and recorded along with the truth files. This data can be analyzed at a later stage to determine any new degradation parameters for *Devanagari*. We hope that an analysis of such real-life data for *Devanagari* would go a long way in determining a more standard technique of identifying script-specific and script-independent simulation parameters.

4.2 Truth Format

To satisfy the requirements outlined in section 3, it was decided to represent the truth in a way that would allow the user to query both word level and glyph level information. Unicode allows us to represent all characters - right from the basic glyphs, to the most complicated combinations of the composite/compound characters possible in the *Devanagari* character set. Though there still exist (rare) characters that cannot be represented using Unicode, it provides a bijective relation between the character code and the compound character in most cases. Representations like transliteration are very elegant, phonetic and more intuitive, but they give an injective relation from the compound character code set to the characters.

The truth will be stored in XML format, which is a very popular markup languages for representing data in an organized fashion. Commercial parsers are available to query collections of XML documents. The use of XML would provide researchers with a means to design parsers and query engines tailored to their specific needs.

5 Truthing tool Design

We have developed a framework (see fig: 1) for truthing based upon our previous experience in designing ergonomic truthing tools, to handle the new challenges posed by *Devanagari* document truthing. An interface for ground-truthing documents typically should be able to display the document image and segments such as words and characters and allow the truther to record the ground-truth of the document segments. The truth should be recorded using a standard representation. It is our intent that *Devanagari* OCR developers will provide results of their system on the standard test set being created, using the same representation as the truth and use the evaluation tools developed to evaluate the performance of their system. This will allow multiple OCRs to be bench-marked and assessed.

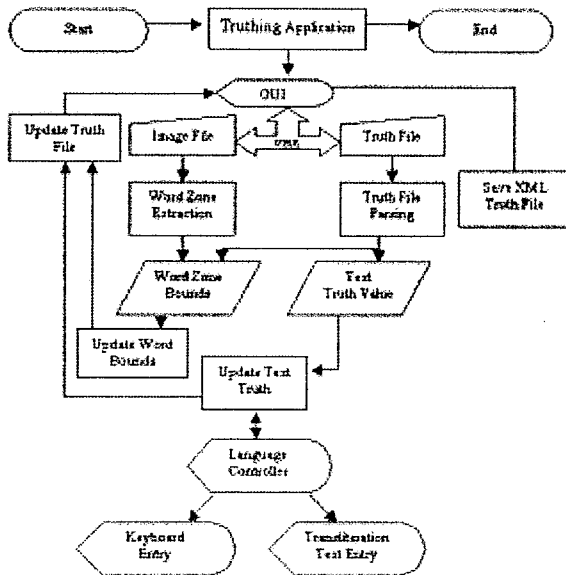


Figure 1: Design of the system

To this end, 400 printed document pages in Hindi and Sanskrit are being scanned. These documents include a variety of printed matter such as pages from old and new books, newspapers, magazines of varying print quality, font types and sizes. Estimating an average of 300 words per document, a data set of 120,000 words would be created.

Unicode has emerged as the de-facto standard for encoding characters of all languages and hence would be the ideal standard to represent both the ground-truth as well as the results of recognition systems. However, while the representation of the truth would be in Unicode the process of generating the truth would require the use of a more intuitive and natural interface.

5.1 Transliteration Scheme - An efficient technique for Devanagari input

To access a character from a font, one needs a way to specify that character in the input source. This is trivial in the case of the Latin alphabet where you can just type **A** on the keyboard when you want to get a typeset **A** in the output and the data can be saved as an ASCII string. In the case of non-Latin character sets, however, one would need either a keyboard designed for the specific script (hard to obtain and difficult to train users) or a transliteration scheme (usually phonetic) that would be able to map ASCII strings to corresponding non-Latin characters. So, to get the Devanagari character क, one could type in the phonetically equivalent (in English) ASCII string **ka**. Since the sounds associated with many Devanagari characters cannot be repre-

sented in phonetic English, approximations such as **RRi** for ऋ are used. Also, many sounds can be phonetically represented in English in more than one way. This has led to people adopting a variety of different transliteration schemes and no single standard has found universal acceptance. However, to the best knowledge of the authors, among the available transliteration schemes, the ITRANS technique supports the most number of languages and hence, was deemed more popular than other schemes. It is also a fairly intuitive scheme that can be learnt quickly.

Using transliteration as a truthing aid enables users to record the truth using a phonetic representation of the *Devanagari* text and the transliterated text can be parsed and converted into the corresponding Unicode for storage. A tool for truthing *Devanagari* documents was designed. Figure 2 shows a screen dump of the truthing tool interface.

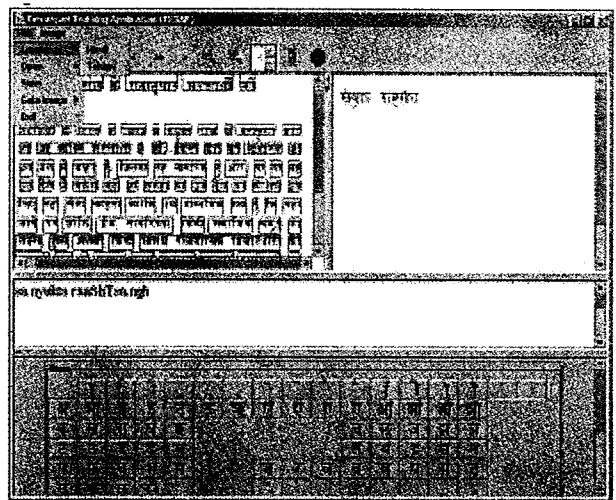


Figure 2: GUI that is used in order to truth the documents

5.2 Image Processing

The tool accepts images in a wide variety of popular image formats including GIF, TIFF and JPEG. Some basic image processing tools such as line and word separation are integrated with the tool. Controls are provided in the tool bar to change thresholds for the word separation. Word boundaries can also be manually selected to correct errors if any, in the automated process. Users can also demarcate non-*Devanagari* or spurious components on the image. Once the words are satisfactorily separated, the image is ready for truthing.

Line separation is based on average pixel intensity (as measured by the grayscale values) in each row. First, sum of the grayscale values along each row

of the image is calculated. Next, the average grey scale value is obtained by dividing this sum with the number of pixels (in that row) whose grey scale value is greater than 0. Regions between 2 successive minima of these intensity values are considered to be the line boundaries. After obtaining line information from the image, each line is considered separately for word separation. The procedure for word separation is very similar to the one outlined for line separation, except that the sum is taken along each column of the line. Spurious lines/words are eliminated by using a predetermined threshold for the height/width. The advantage of this method (over simple summing of the grayscale values along each row/column) is that it produces much sharper histogram profiles in the presence of uniform background noise.

The keyer view the segmented document image in the box at the top left of the screen. The truth can either be input using the simulated keyboard in the bottom panel or by typing in transliterated text using the ITRANS convention. The truth is stored along with the word co-ordinates in Unicode format as described earlier. The truth will be rendered from the Unicode values using a *Devanagari* font in the top right panel to give the keyer visual feedback so that the keyer can compare the truth and the *Devanagari* text side-by-side. The current word being truthed is highlighted both in the image as well as in the rendered truth.

Representing the truth in a standard format such as Unicode allows results and truth to be easily compared. We also propose to provide query tools to retrieve word images based on a search string in transliterated text as well as Unicode values. The database would initially contain word images tagged with the corresponding word truth in Unicode format and also the transliterated text for that word in ITRANS format. We also propose to make available the truth files in XML format to enable users to design customized parsers and other tools for analyzing the data. Unicode uses a hexadecimal code for representation of the basic character, and the modifiers. Detailed rules exist to generate characters with variations like the virama. compound characters, characters with modifiers etc [15].

6 Conclusions and future work

The initial datasets, consisting of documents with their word-level truth have been made freely available to the public in early 2003. Distribution via the world wide web is being planned. The truthing tool and query engines will also be made available to the public. We would like to acknowledge the assistance provided by Prof. B.B.Chaudhuri and U. Pal of the Indian Statistical Institute, Calcutta in the design of the composition of the data sets.

The following data resources will also be created during the course of this effort.

- Create standard training and testing sets of machine-printed and handwritten . *Devanagari* characters for training and evaluation of character recognizers with ground-truth.
- Create and/or consolidate any existing machine readable dictionaries for Sanskrit and Hindi words that can be used for various Natural Language Processing (NLP) applications.
- Create and/or consolidate any machine readable corpora of texts for Sanskrit and Hindi that can be used for various NLP applications.
- Develop software for grammatical tagging of corpora for use in various NLP applications.

References

- [1] Hanhong Xue, Venu Govindaraju *On the Dependence of Handwritten Word Recognizers on Lexicons*. IEEE Pattern Analysis and Machine Intelligence. Vol 24, No 12. pp. 1553-1564.
- [2] J. J. Hull. *A database for handwritten text recognition research*. IEEE Transaction on Pattern Analysis and Machine Intelligence, 16(5):550-554, May 1994.
- [3] Isabelle Guyon, Robert M. Haralick, et al. *Data Sets For OCR And Document Image Understanding Research* Proceedings of the 2nd International Conference on Document Analysis and Recognition.
- [4] Haralick Robert M *UW-II English/Japanese Document Image Database*. Intelligent Systems Laboratory, University of Washington.
- [5] Schlosser Steven G. *ERIM Arabic Database* Document Processing Research Program. Information and Materials Applications Laboratory, Environmental Research Institute of Michigan.
- [6] *Japanese Character Image Database* Center for Excellence in Document Analysis and Research,
- [7] D.H. Kim, Y.S. Hwang, S.T. Park, et al. *Handwritten Korean character image database* Proceedings of the 2nd International Conference on Document Analysis and Recognition.
- [8] Veena Bansal, R.M.K. Sinha. *On How to Describe Shapes of Devanagari Characters and Use Them for Recognition*. Fifth International Conference on Document Analysis and Recognition.

- [9] Sethi, I.K., Chatterjee, B., *Machine Recognition of Constrained Hand Printed Devanagari*. Pattern Recognition(9), 1977, pp. 69-75.
- [10] R.M.K. Sinha, Mahabala H. *Recognition of Devanagari Script*. IEEE Trans on Systems, Man, and Cybernetics, Vol. 9, pp 435-441, 1979.
- [11] A. Bishnu, B.B. Chaudhuri. *Segmentation of Bangla Handwritten Text into Characters by Recursive Contour Following* Fifth International Conference on Document Analysis and Recognition.
- [12] J.C. Perez-Cortes, J.C. Amengual, J. Arlandis, and R. Llobet. *Stochastic error correcting parsing for ocr post-processing*. In *International Conference on Pattern Recognition ICPR-2000*, Barcelona, Spain.
- [13] *Online Database of languages: www.ethnologue.com*
- [14] Chaudhuri, B.B.; Pal, U. *An OCR system to read two Indian language scripts: Bangla and Devanagari*. Proceedings of the 4th International Conference on Document Analysis and Recognition (ICDAR '97) Volume: 2 , 1997 Page(s): 1011 -1015 vol.2
- [15] *The Unicode Standard Version 3.0*. Addison-Wesley, 2000. ISBN 0-201-61633-5.
- [16] S. Setlur and V. Govindaraju, *Translingual OCR by Template Correlations* Proceedings of SPIE, Vol. 3964, 1999, pp. 305-312.
- [17] S. Setlur, A. Lawson, V. Govindaraju, and S. Srihari, *Large scale address recognition systems - Truthing, testing, tools and other evaluation issues*, International Journal of Document Analysis and Recognition, Vol. 4, No. 3, March 2002, pp. 154-169.
- [18] V. Bansal, R.M.K. Sinha, *Integrating Knowledge Sources in Devanagari Text Recognition*, IEEE Transactions on Systems, Man and Cybernetics, Vol. 30, No. 4, pp. 500-505, 2000.
- [19] B. Chaudhuri and U. Pal, *A complete printed Bangala OCR system*, Journal of Pattern Recognition, Vol. 31, No. 5, pp. 531-549, 1998.
- [20] T. K. Ho and Henry S. Baird. *Large-Scale Simulation Studies in Image Pattern Recognition* IEEE Trans. PAMI, Vol. 19, No. 10, pp. 1067-1079, October 1997.
- [21] Henry S. Baird, T. Kanungo, R. M. Haralick. *Validation and Estimation of Document Degradation Models* Proc., 4th UNLV Symp. on Document Analysis and Information Retrieval, Las Vegas, Nevada, April 24-26, 1995.
- [22] D.-S. Lee, C. Nohl, Henry S. Baird, *Language Identification in Complex, Unoriented, and Degraded Document Images*. Proc., IAPR 1996 Workshop on Document Analysis Systems, Malvern, PA, October 14-16, 1996.
- [23] T. Kanungo and P. Resnik, *The Bible, Truth, and Multilingual OCR Evaluation*. Proc. of SPIE Conference on Document Recognition and Retrieval (VI), vol. 3651, San Jose, CA, January 27-28, 1999
- [24] Bansal, Veena and R.M.K. Sinha *Segmentation of touching and fused Devanagari characters* Pattern Recognition, volume 35 (2002), number 4 pp. 875-893.
- [25] Oivind Due Trier, Anil K Jain, Torfinn Taxt, *Feature Extraction Methods For Character Recognition A Survey (1995)* Pattern Recognition Vol 29, No. 4 pp. 641-662, 1996.
- [26] U. Garain, B.B. Chaudhuri, *Segmentation of Touching Characters in Printed Devnagari and Bangla Scripts Using Fuzzy Multifactorial Analysis*. Proceedings of the 6th International Conference on Document Analysis and Recognition. (ICDAR '01).

Multilingual OCR Ground Truth from Printed and Web Sources

Mark **Yuliya** **Kristen**
Turner **Katsnelson** **Summers**
Vredenburg Co.
4831 Walden La., Lanham, MD 20706
{mturner, ykatsnelson,
ksummers}@vredenburg.com

We describe the construction of a multilingual ground truth for OCR, using both hard-copy and electronic documents as sources. Languages include those with alphabetic and ideographic scripts. The goal was to produce a ground truth with a variety of fonts, font sizes, image characteristics, and degradations. For both sources, the content was organized into single zones for OCR training.

Hard copy was gathered from a variety of sources: newspapers, magazines, books, flyers, etc. Pages of the hard copy documents were scanned as bitonal 300 dpi TIF images and key-entered to produce Unicode (UTF-8) text; the key-entry preserved the line boundaries of the TIF images.

From electronic (HTML) documents, the foreign language text was stripped out, preserving end-of-paragraph marks, and the text was used to generate MS Word documents. A number of classes were created to represent combinations of fonts and font sizes. Each of these MS Word documents was assigned to one of these font/size classes and manually edited to apply the font and font size of that class. Rendering these documents yielded a set of 300 dpi ideal synthetic images.

We produced degradation by first physically scanning and faxing the documents. Documents were faxed to a digital fax machine, faxed to a plain paper fax and rescanned at 300 dpi, and scanned at 300 dpi. From each of the font/font size variations in the electronic document collections, we chose a fixed number of images to fax and scan. After scanning and faxing, each resulting fax was combined with a set of fax noise templates and each scanned document combined with a set of scanner noise templates.

By combining hard copy and electronic sources, and applying degradations, we are able to produce an OCR training corpus with realistic assortment of fonts, font sizes, and image characteristics.

Ground Truth Data for Document Image Analysis

Glenn Ford George R. Thoma

Lister Hill National Center for Biomedical Communications
National Library of Medicine
Bethesda, Maryland

Abstract

The ground truth data described here is collected from the production operation of MARS (Medical Article Records System), a system combining scanning, OCR, document image analysis and lexical analysis techniques. Developed by an R&D division of the National Library of Medicine (NLM), MARS automatically extracts bibliographic data from paper-based biomedical journals to populate the Library's flagship database, MEDLINE®, used worldwide by biomedical researchers and clinicians. The bibliographic data extracted include the article title, author names, institutional affiliations and abstracts.

This ground truth data includes document images, OCR output and operator-verified data at the page, zone, line, word, and character levels. It is accessible online via a public website to enable researchers to develop innovative and efficient algorithms for automatic zoning (page segmentation), labeling (field identification), lexical analysis techniques to correct OCR errors, and techniques for reformatting syntax to adhere to established conventions. In addition, we offer a tool (Rover) to visually compare the results of such programs to the ground truth data. The ground truth and results data are in XML, and Rover is written in Java. The overall website development uses MacroMedia Dreamweaver UltradDev 4 to provide a rich interface and extensive database connectivity.

1 Introduction and objective

Research in document image analysis is greatly dependent on ground truth data for the design, training and testing of algorithms for data identification and extraction. However, ground truth datasets and their associated analysis and visualization tools are usually created to analyze problems in specific applications and datatypes: skewed document images (Okun et al.)¹, handwritten documents (Cha and Srihari)², video sequences (Doermann and Mihalcik)³, statistical data (Swayne et al.)⁴, and speech signals (Barras et al.)⁵. Moreover, apart from the domain-specific nature of these datasets and tools, they are usually limited as to

operating platforms and data formats, as described in an excellent taxonomy on this subject by Kanungo et al.⁶

To our knowledge no ground truth dataset exists that represents the corpus of biomedical journals, and none with the goal of extracting the text representing the bibliographic fields descriptive of the articles within these journals. Such bibliographic data is extracted automatically from scanned biomedical journals by the Medical Article Records System (MARS) built and operated at the National Library of Medicine to populate its flagship database, MEDLINE®. MARS relies on rule-based algorithms for automatic zoning, labeling and reformatting.⁷⁻⁹

In the course of normal production operations, MARS generates vast amounts of document images and OCR-converted and operator-verified data. This data, as ground truth, would be invaluable in the design of innovative and efficient algorithms for automatic zoning, labeling and reformatting by the computer science and medical informatics communities. To aid in this effort, we are developing MARG (*Medical Article Records Groundtruth*), a ground truth database accessible via the Web.

This ground truth data includes page, zone, line, word, and character level information. In addition to providing a public site for researchers worldwide to develop and test their algorithms, we propose a tool to enable them to graphically visualize the ground truth data and employ an automated analysis assistant. Code-named Rover (*gROundtruth Vs. Engineered Results*), this automated analysis assistant will compare the results of a researcher's program to the ground truth data. The ground truth and results data are in XML, and Rover will be written in Java. The overall website development uses MacroMedia Dreamweaver UltradDev 4 to provide a rich interface and extensive database connectivity.

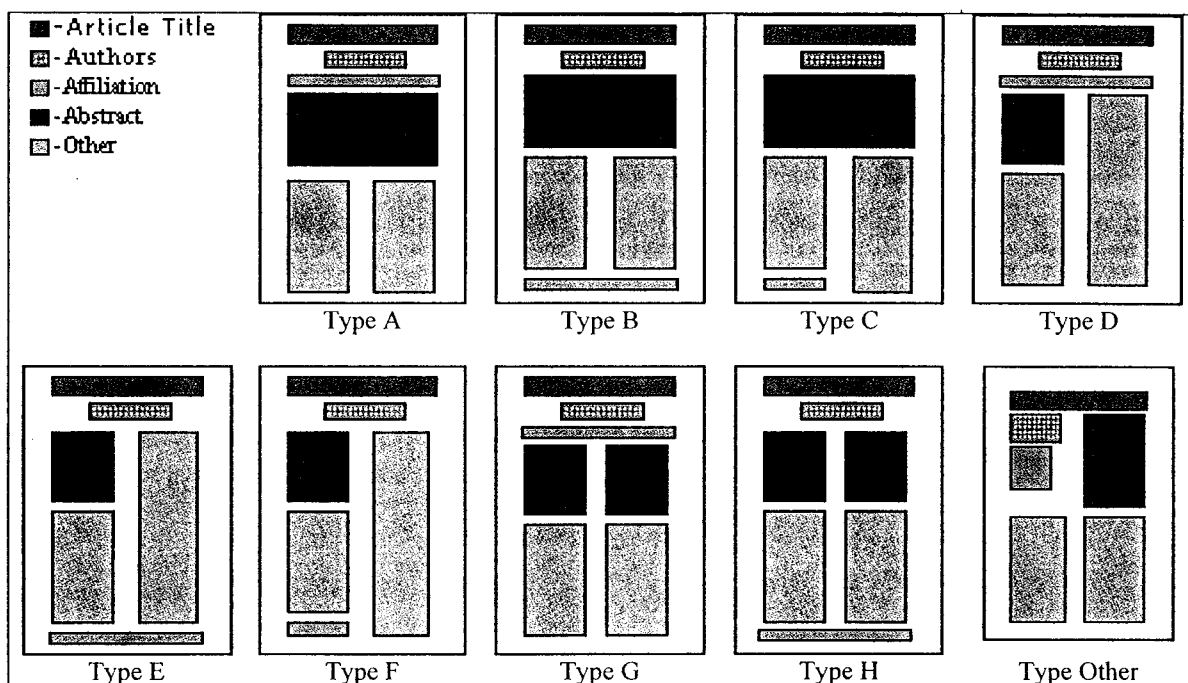


Figure 1 Page Layout Types

2 Design considerations

2.1 Page layout

Identifying geometric features to design algorithms for automated data extraction is a non-trivial task since there is a variety of layout geometries in the 4,300 journal titles indexed in MEDLINE, though most follow the reading order paradigm of article title-author names-affiliation-abstract. Ground truth data representative of the corpus of biomedical journals must therefore include samples of all significant layout types.

The MARG site contains examples of 9 layout types (Figure 1) grouped according to the placement of the Article Title, Author(s), Affiliation, and Abstract. The general description each type is as follows:

- Type A - The Title, Author, Affiliation and Abstract appear in the defined order and are located in the upper half of the page.
- Type B - The Title, Author and Abstract are in the upper portion of the page. The Affiliation is located at the bottom.
- Type C - The Title, Author and Abstract are in the upper half of the page. The Affiliation is
- Type D - The Title, Author, and Affiliation are in the upper half of the page. The Abstract usually is in the first column. Variations include the abstract continuing into a portion of the second column.
- Type E - The Title, Author, and Affiliation are in the upper half of the page. The Abstract is single-columned, but above the body text of the article in most cases.
- Type F- The Title and Author are in the upper half of the page. The Affiliation is along the bottom-left. The Abstract usually is in all or some of the first column. Variations include the abstract continuing into a portion of column 2.
- Type G - The Title, Author and Affiliation are in the upper half of the page. The Abstract is in two adjacent columns.
- Type H - The Title and Author are in the upper half of the page. The Affiliation is along the

single columned and located in the left column of double column text. Variations include: body of text below the double column affiliation area that is material not relevant to MEDLINE citations.

bottom. The Abstract is double columned, but above the body text of the article in most cases.

- Type Other - This category, holding all unusual layouts encountered, account for approximately 23% of the journal collection. Layouts in this category have not been categorized further as yet.

2.2 Distribution of page layouts

Once the layouts were classified in the types shown in Section 2.1, it was of interest to determine their frequency of occurrence. Figure 2 shows a distribution of these defined types.

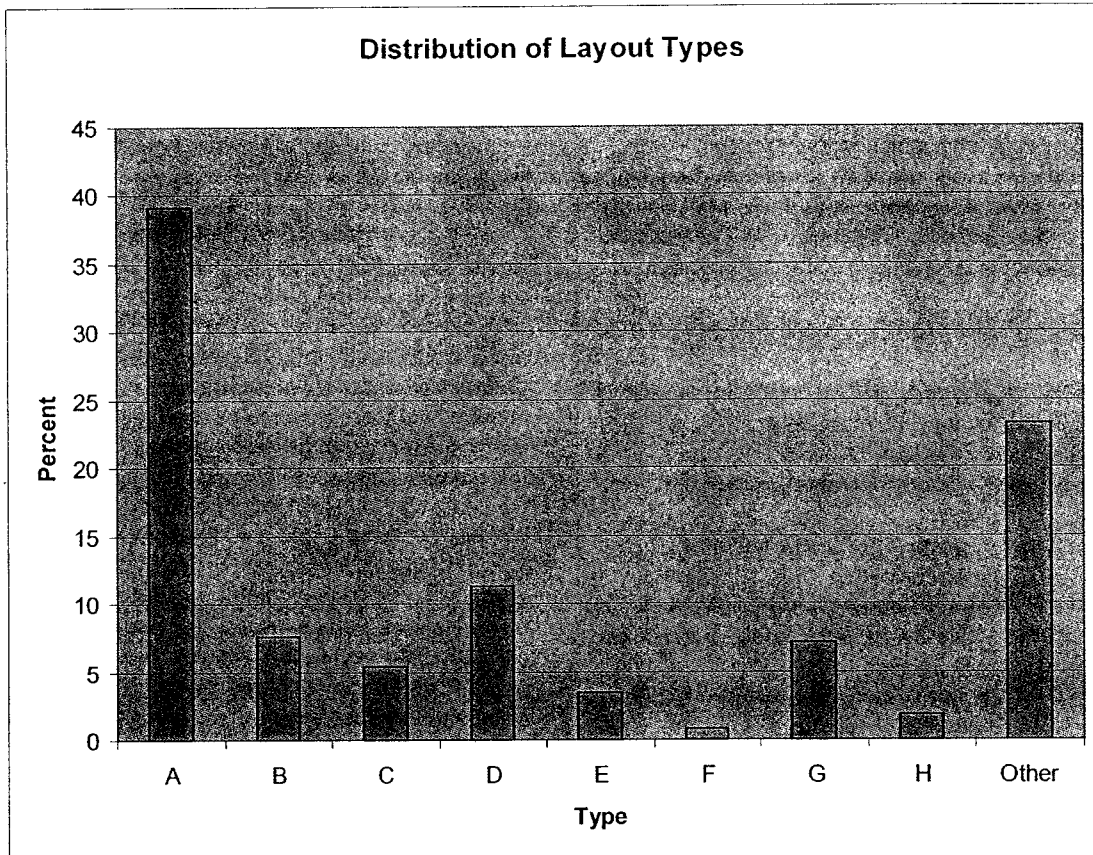


Figure 2 Distribution of layout types

2.3 Data format

The format chosen for all data is XML: for images, OCR-converted data and operator-verified data since XML excels in adaptation (to accommodate changes in data), maintenance, linking (from one piece of data to another), simplicity, and portability over networks, operating systems and development environments¹⁰⁻¹².

2.4 Modifying existing data

Despite its undeniable value, the existing data has certain deficiencies. For example, although OCR output characters in error are corrected as part of the text verification stage in production, their attributes (e.g., *italics* or **bold**) are not. But these attributes can serve as features to create algorithmic rules to correctly identify zones or labels. Attribute information will be included in subsequent releases of the ground truth dataset.

3 Rover: a visualization and analysis tool

Once the data is available in XML format along with the original TIFF image, researchers need the functionality to visualize and analyze the data. Specifically, the following functions are needed:

- Load a TIFF image and the corresponding XML file into an application, and display the XML data, where appropriate, overlaid on the image.
- Modify and add new ground truth data.
- Compare the results of new algorithms against the ground truth data.

The first two functions are provided by TrueViz, described in a survey of visualization tools⁶. TrueViz is a public domain tool developed at the University of Maryland for visualizing, creating and editing ground truth and metadata. It is implemented in Java and works on Windows and Unix platforms (specifically, it has been successfully tested in the Windows 2000 and Sun Solaris 2.6 environments). It reads and stores ground truth and metadata in XML format, and reads the corresponding TIFF images. It allows the user to inspect ground truth data at many levels, viz., at the page, zone, line, word, and character levels, and provides pertinent information at each level. For example, at the character level, such information includes the character code, font type and style, and bounding box (x1,y1,x2,y2) coordinates.

TrueViz is a suitable platform to initiate the design of Rover, an analysis assistant that will offer all three functions mentioned above. To serve as an effective analysis assistant, Rover will extend TrueViz to provide researchers the capability to *compare* their XML data to ground truth XML data graphically (Figure 3), and thereby help them iteratively refine their algorithms. In addition, Rover will provide statistics and visual presentations on specified areas. For example, the user would be able to use Rover to compare all characters in the dataset that are **bold**, and to enumerate errors. Rover would visually locate the mistakes and report statistics on the query. In this example it would report the number of bold characters, the number detected correctly, and the number detected incorrectly, both as absolute numbers and as percentages. Rover would also export this information to a database or spreadsheet for further analysis.

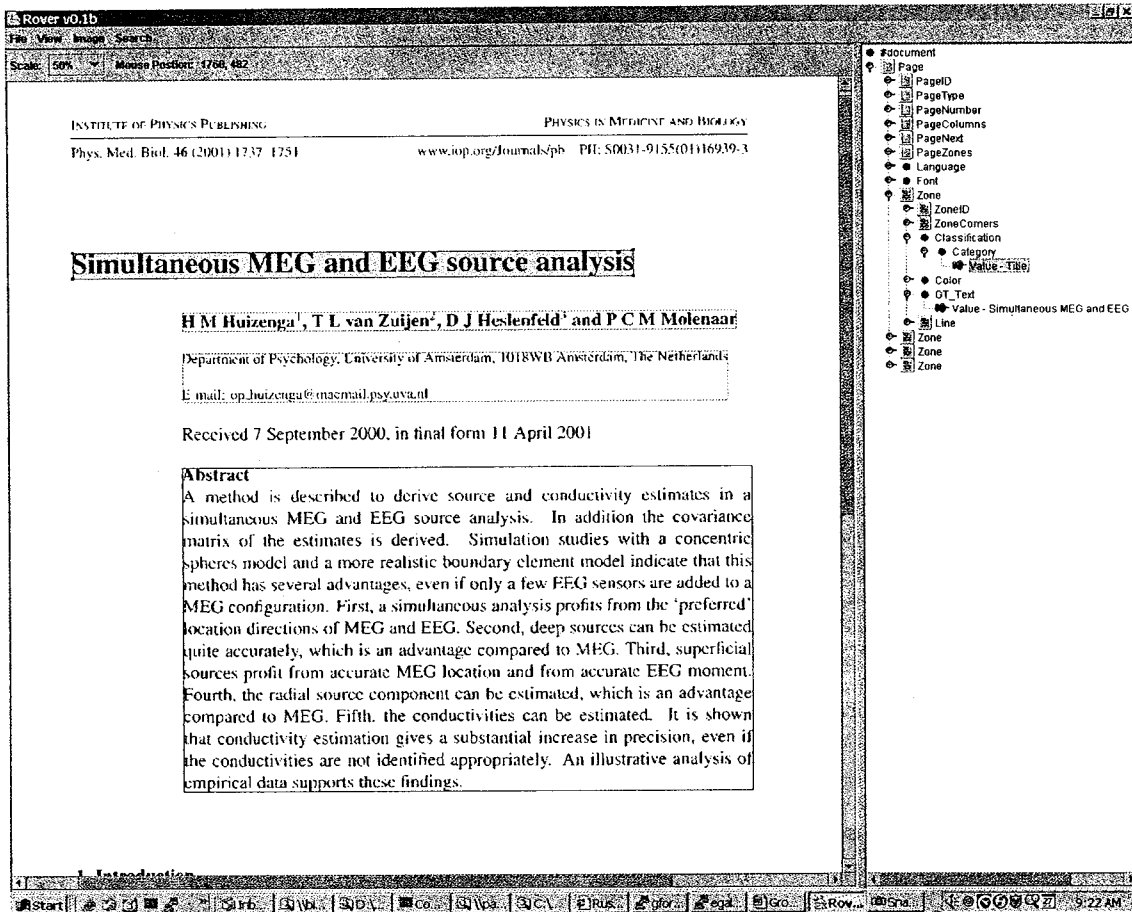


Figure 3 Rover screen snapshot of TIFF page and zone segments.

4 Ground truth website

A website provides access to the ground truth data, though it will go beyond serving as a simple data repository. Our objective is to encourage researchers to share and exchange ideas, as well as provide feedback

to our development team for new desirable features. Figure 4 shows the organization of the website and how the elements interconnect. This section presents an overview of the website layout and functionality.

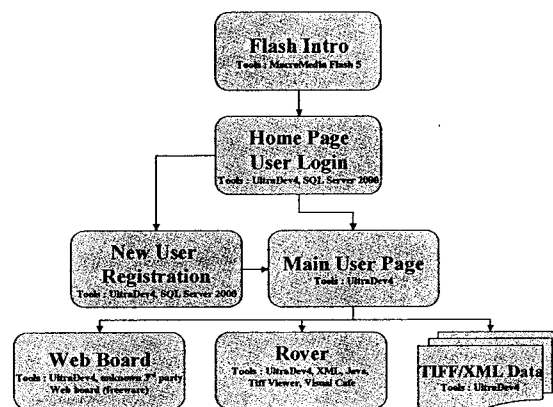


Figure 4 Main sections of website and tools

4.1 Product introduction

A Macromedia Quick Flash "movie" is displayed to the users, though this may be bypassed. The metaphor used is a Sherlock Holmes style magnifying glass moving over a rare biomedical document.

4.2 User login

On the main page of the website, users are asked for a name and password. This data is stored in a SQL Server 2000 database along with the website data. We use Macromedia Ultradev as the design tool, because it readily allows for password protection and database connectivity. While we do not anticipate restricting anyone from accessing this site, we intend to require all users to register. When registered users log in, they will be taken to the main user page. Unregistered users will be presented with a registration page before being allowed access.

4.3 New user registration

The system will require first time users to enter a few important items of information that will enable us to provide security to the system, access to the Bulletin Board, and provide the ability to contact users with important new data or features added to the website in the future. The registration page was developed using Ultradev, which allows the developer a graphical design environment and provides rapid page development.

4.4 Data access

Once logged in or registered, the user would have access to all the ground truth data and tools. Since the data collection will be quite large, the system will allow users to download the entire data set or any subset they choose. For example, some users may only be interested in certain types of journal layouts, such as double column abstracts.

4.5 Data analysis

The users will be provided a link to launch Rover. While the initial version of this tool will possess the current functionality of TrueViz, the complete analysis support discussed earlier will be designed in a later phase.

4.6 Bulletin board

This is for the user community to report bugs, and use as a forum for discussions related to algorithmic development. Here the users may also upload and download files, such as technical papers written, algorithm source code, and new ground truth data.

5 Summary

We describe here a system for the distribution of ground truth data collected from the production operation of MARS, a system for the automated extraction of bibliographic data from scanned biomedical journals. This ground truth data includes document images, OCR output and operator-verified data at the page, zone, line, word, and character levels. It is accessible online via a public website to enable researchers to develop innovative and efficient algorithms for automatic zoning (page segmentation), labeling (field identification), lexical analysis techniques to correct OCR errors, and techniques for reformatting syntax to adhere to established conventions.

References

1. Okun O, et al. An experimental tool for generating ground truths for skewed page images. Proc. SPIE Document Recognition and Retrieval VIII. Vol. 4307, San Jose CA, January 2001, 22-33.
2. Cha S-H, Srihari SN. Handwritten document image database construction and retrieval system. Proc. SPIE Document Recognition and Retrieval VIII. Vol. 4307, San Jose CA, January 2001, 13-21.
3. Doermann D, Mihalcik D. Tools and techniques for video performance evaluation. Proc. 15th International Conference on Pattern Recognition. Barcelona, Spain, September 2000, 167-70.
4. Swayne DF et al. Xgobi: interactive dynamic data visualization in the X window system. Journal of Computational and Graphical Statistics 7, 1998.
5. Barras C, et al. Transcriber: a free tool for segmenting, labeling and transcribing speech. Proc. 1st Int. Conf. Language Resources and Evaluation. Granada, Spain, May 1998; 1373-76.
6. Kanungo T, et al. TRUEVIZ: A groundtruth/metadata editing and visualizing toolkit for OCR. Proc. SPIE Document Recognition and Retrieval VIII. Vol. 4307, San Jose CA, January 2001, 1-12.
7. Thoma GR. Automating the production of bibliographic records for MEDLINE. Internal R&D report, CEB, LHCBC, NLM; September 2001; 92pp. Available: archive.nlm.nih.gov/pubs/biblio/biblio.php
8. Thoma GR. Document image analysis and understanding R&D. Internal R&D report to the Board of Scientific Counselors, CEB, LHCBC, NLM;

October 2001; 55pp. Available:
archive.nlm.nih.gov/pubs/biblio/biblio.php

9. Kim J, Le DX, Thoma GR. Automated Labeling in Document Images. Proc. SPIE, Vol. 4307, Document Recognition and Retrieval VIII, San Jose CA, January 2001, 111-22.

10. Desai G and Fenner J. Unleash the power of XML. Imaging and Document Solutions, Vol. 9, No. 12, Dec 2000, 29-32.

11. Pearson G, Moon CW. Bridging two biomedical journal databases with XML: A case study. Proc. 14th IEEE Symposium on Computer-Based Medical Systems. Los Alamitos, CA: IEEE Computer Society. July 2001, 309-14.

12. World Wide Web Consortium, XML Working Group, "XML 1.0 Specification 10-February-1998", ed. T. Bray, J. Paoli, C.M. Sperberg-McQueen. Available in various forms: www.w3.org/TR/1998/REC-xml-19980210.

Page Structure 2

High Performance Document Layout Analysis

Thomas M. Breuel
PARC, Palo Alto, CA, USA
tmb@parc.com

Abstract

In this paper¹, I summarize research in document layout analysis carried out over the last few years in our laboratory. Correct document layout analysis is a key step in document capture conversions into electronic formats, optical character recognition (OCR), information retrieval from scanned documents, appearance-based document retrieval, and re-formatting of documents for on-screen display. We have developed a number of novel geometric algorithms and statistical methods. Layout analysis systems built from these algorithms are applicable to a wide variety of languages and layouts, and have proven to be robust to the presence of noise and spurious features in a page image. The system itself consists of reusable and independent software modules that can be reconfigured to be adapted to different languages and applications. Currently, we are using them for electronic book and document capture applications. If there is commercial or government demand, we are interested in adapting these tools to information retrieval and intelligence applications.

1 Introduction

Document layout analysis is a key step in converting document images into electronic form. Document layout analysis identifies key parts of a document, like titles, abstracts, sections, page numbering, and puts the text on a page into the correct reading order, which is a prerequisite for optical character recognition (OCR), as well as most forms of document retrieval.

Traditional document layout analysis methods will generally first attempt to perform a complete global segmentation of the document into distinct geometric regions corresponding to entities like columns, headings, and paragraphs using features

like proximity, texture, or whitespace. Segmentation into regions are often carried out using heuristic methods based on morphology or “smearing” based approaches, projection profiles (recursive X-Y cuts), texture-based analysis, analysis of the background structure, and others (for a review and references, see [7]). Each individual region is then considered separately for tasks like text line finding and OCR. The problem with this approach lies in the fact that obtaining a complete and reliable segmentation of a document into separate regions is difficult to achieve in general. Some decisions about which regions to combine may well involve semantic constraints on the output of an OCR system. However, in order to be able to pass the document to the OCR system in the first place, we must already have identified text lines, leading to circular dependencies among the processing steps.

In our work, we use exact and globally optimal geometric algorithms, combined with robust statistical models, to model and analyze the layout of pages. By combining these algorithms carefully, we arrive at an overall approach to document layout analysis that avoid the circular dependencies of traditional methods and greatly reduces the number of parameters needed to “tune the system”. The resulting document layout analysis systems are *applicable to a wide variety of languages and layouts*, and have proven to be *robust to the presence of noise* and spurious features in a page image. Below, we will first examine the individual steps and algorithms needed for this approach to document layout analysis and then describe how these algorithms are put together into an overall document layout analysis system. It can be accomplished reliably using the whitespace analysis algorithm described in this paper using a novel evaluation function.

¹This paper is a compilation of results, figures, and descriptions from a number of previously published papers. Please see the individual sections for attributions and references.

Keeping up on patents

Microsoft's White Paper on patent law is available on CD-ROM. The CD-ROM contains a copy of the White Paper, a list of patent attorneys, and a list of patent attorneys who are available to provide legal services to small businesses. The CD-ROM also includes a copy of the White Paper on patent law, a list of patent attorneys, and a list of patent attorneys who are available to provide legal services to small businesses.

Abstracts on CD-ROM now an option

The Institute for Scientific Information (ISI) has announced a new service for its ISI/BIOMED database. The new service, called ISI/BIOMED Abstracts, provides abstracts for all articles in the ISI/BIOMED database. The abstracts are available in both print and electronic form. The abstracts are available in both print and electronic form.

Build your own nucleus with a mouse

Version 3.0 of FCC Modem Software for Windows is now available. The software allows you to build your own nucleus with a mouse. The software is available for Windows 3.11 and Windows 95. The software is available for Windows 3.11 and Windows 95.

Workstations, workstations, workstations

Parallel Unix on 68000. The software allows you to run Unix on a 68000 processor. The software is available for Windows 3.11 and Windows 95. The software is available for Windows 3.11 and Windows 95.

Biomedical Telemetry: The Formative Years

By R. STUART MACLEAY

Biomedical telemetry is a rapidly growing field. It involves the transmission of physiological data from a patient to a remote location. The data is used for diagnosis and treatment. The field is growing rapidly and is expected to continue to grow in the future.

The field of biomedical telemetry is a rapidly growing field. It involves the transmission of physiological data from a patient to a remote location. The data is used for diagnosis and treatment. The field is growing rapidly and is expected to continue to grow in the future.

Overview of the ITRF Conceptual Design Activities

R. Tomlinson

The conceptual design of the ITRF (International Telemetry Relay Facility) is a complex task. It involves the design of a system that can receive and transmit data from a satellite. The system is designed to be reliable and to have a long life span. The system is designed to be reliable and to have a long life span.

The conceptual design of the ITRF (International Telemetry Relay Facility) is a complex task. It involves the design of a system that can receive and transmit data from a satellite. The system is designed to be reliable and to have a long life span. The system is designed to be reliable and to have a long life span.

Figure 1: Examples of the result of whitespace evaluation for the detection of column boundaries in documents with complex layouts (documents A00C, D050, and E002 from the UW3 database).

Figure 1 shows three examples of document layouts with column boundaries highlighted. Each diagram includes a grid and a list of column separators. The diagrams illustrate how whitespace evaluation can be used to detect column boundaries in documents with complex layouts.

Figure 1: Examples of the result of whitespace evaluation for the detection of column boundaries in documents with complex layouts (documents A00C, D050, and E002 from the UW3 database). Note that even complex layouts are described by a small collection of column separators.

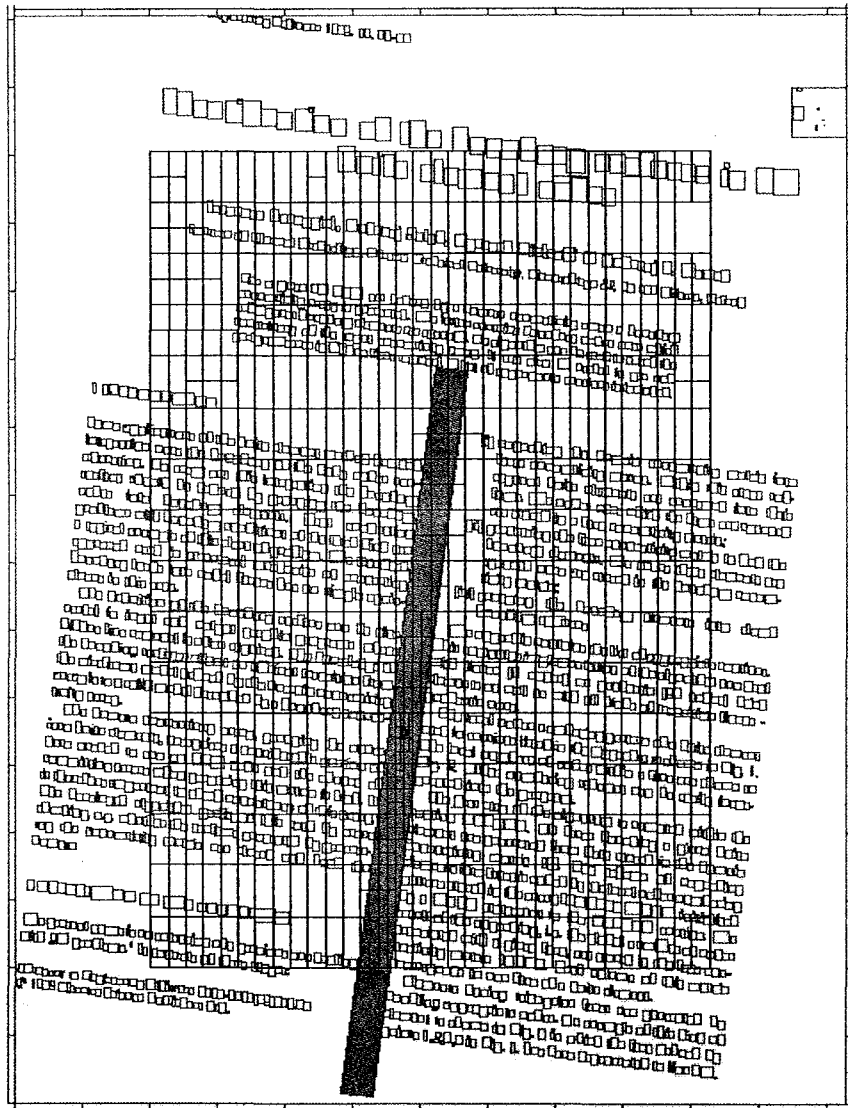


Figure 2: Globally optimal whitespace finding using non-axis aligned rectangles. The red rectangles in the background are bounding boxes for connected components in a scanned document. The large, non-aligned slender rectangle in the center is the column boundary found by the algorithm. Overlaid is a grid showing the hierarchical exploration of the parameter space carried out by the algorithm.

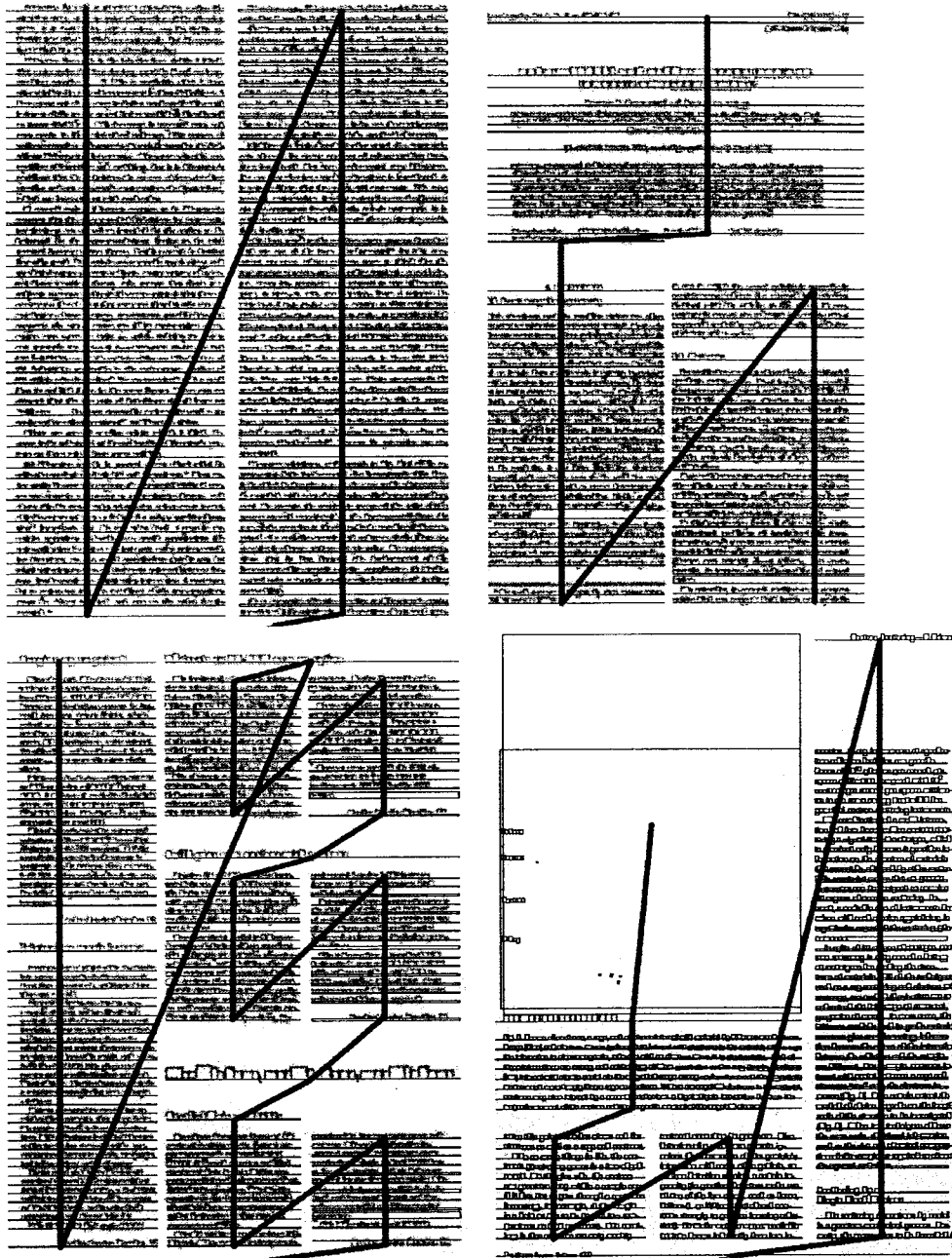
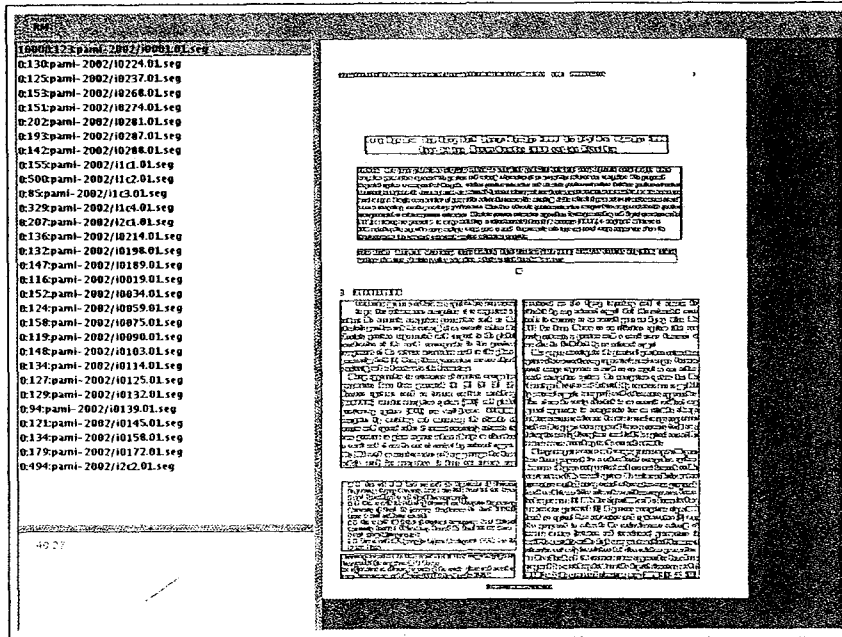
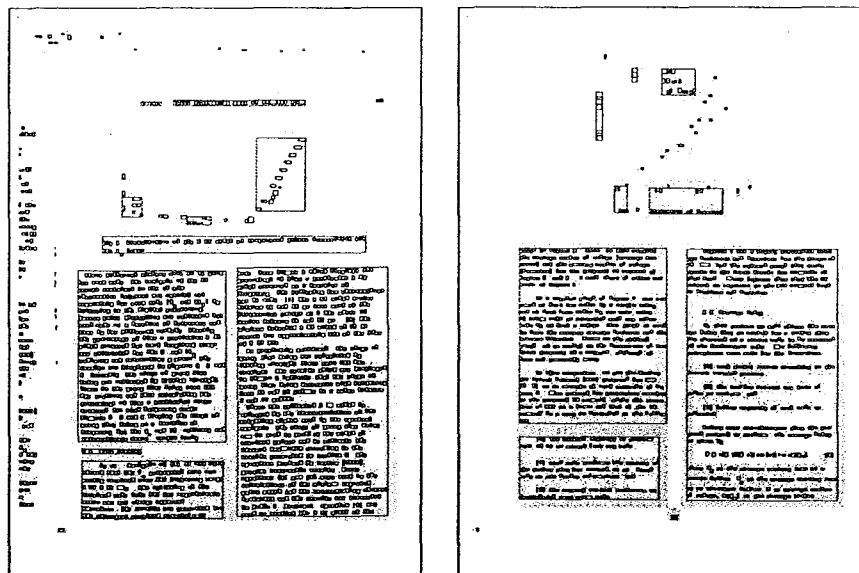


Figure 3: Recovering reading order by topological sorting; the thin black horizontal lines indicate text line segments, and the thick black lines running down and diagonally across the image indicate reading order; they connect the center of each text line with the center of the text line immediately following it in reading order. Text lines used are the lines as returned by the constrained text line finder; that is, text lines extend all the way across each column, even if actual characters only fill the line partially. Note that floating elements like headers and footers were not removed prior to determination of reading order and simply appear at some point inside the reading order (see the text for details).



(a)



(b)

Figure 4: Scale-space layout analysis and appearance based document retrieval. (a) Interactive GUI permitting fast manually supervised segmentation of document layouts using hierarchical layout analysis. By permitting the user to select interactively, with visual feedback, a scale at which the page is to be segmented, most documents can be segmented within a few seconds. This greatly speeds up manual layout analysis relative to approaches that require the user to mark each layout block individually. (b) An example of appearance-based retrieval. The database consists of 751 documents from the UW-1 collection. The query document is shown on the left and its closest match in the database is shown on the right. The appearance-based retrieval system adjusts the scale at which the document is segmented automatically while documents are being matched. This greatly improves the accuracy of appearance based retrieval and reduces the need for manual corrections to layouts.

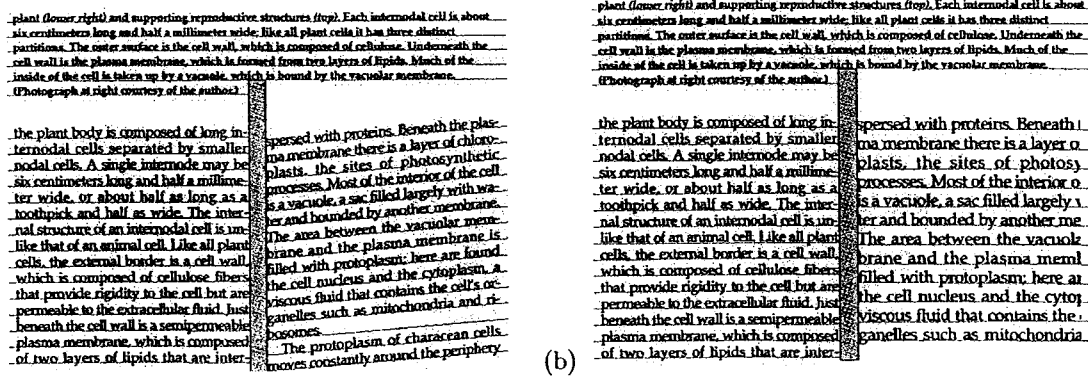


Figure 5: Application of the constrained line finding algorithm to simulated variants of a page. Gutters (obstacles) were found automatically using the algorithm described in the paper and are shown in green. Text lines were found using the constrained line finder and are shown in faint red. (a) Two neighboring columns have different orientations (this often occurs on the two sides of a spine of a scanned book). (b) Two neighboring columns have different font sizes and, as a result, the baselines do not line up.

2 Fast and Simple Maximum Empty Rectangles

A key step in document layout analysis is the determination of the structure of the page background. This structure allows us to identify major page layout features like columns and sections.

Background structure analysis as an approach to document layout analysis has been described by a number of authors [1, 12]. The work by Baird *et al.* [2] analyzes background structure in terms of rectangular covers, a computationally convenient and compact representation of the background. However, past algorithms for computing such rectangular covers have been fairly difficult to implement, requiring a number of geometric data structures and dealing with special cases that arise during the sweep (Baird, personal communication). This has probably limited the widespread adoption of such methods despite the attractive properties that rectangular covers possess. Our new algorithm requires no geometric data structures to be implemented and no special cases to be considered; it can be expressed in less than 100 lines of Java code. In contrast to previous methods, it also returns solutions in best-first order.

We define the maximal white rectangle problem as follows. Assume that we are given a collection of rectangles $C = \{r_0, \dots, r_n\}$ in the plane, all contained within some given bounding rectangle r_b . In layout analysis, the r_i will usually correspond to the bounding boxes of connected components on the page, and the overall bounding rectangle r_b will represent the whole page. Also, assume that we are given an evaluation function for rectangles $Q : \mathbb{R}^4 \rightarrow \mathbb{R}$; in the simplest case, this will simply be the area, although we will consider more complex evaluation functions in the next sec-

tion. The maximal white rectangle problem is to find a rectangle $\hat{r} \subseteq r_b$ that maximizes $Q(\hat{r})$ among all the possible rectangles $r \subseteq r_b$, where r overlaps none of the rectangles in C . The key idea behind the algorithm is similar to quicksort or branch-and-bound methods; details can be found in the references [3].

An application of this algorithm for finding a greedy covering of a document from the UW3 database with maximal empty rectangles is shown in Figure 1. Computation times for commonly occurring parameter settings using a C++ implementation of the algorithm on a 400MHz laptop are under a second. As it is, this algorithm could be used as a drop-in replacement for the whitespace cover algorithm used by [11], and it should be useful to anyone interested in implementing that kind of page segmentation system. However, below, this paper describes an alternative use of the algorithm that uses different evaluation criteria.

3 Improved Whitespace Evaluation

As we can see in Figure 6, merely computing an optimal whitespace cover does not necessarily yield a meaningful analysis of the page background. The approach by [1] addresses this problem by constructing an evaluation function Q of candidate whitespace rectangles based on their size and aspect ratio. However, that information by itself is not very reliable for distinguishing meaningful whitespace components from meaningless ones.

We have developed a simple set of evaluation criteria that identifies meaningful whitespace with an estimated error rate of less than 0.5% on the UW3 database with a single set of parameters. The idea is that for layout whitespace to be meaningful, it should separate text. Therefore, we require rectangles returned by the whitespace analysis algorithm

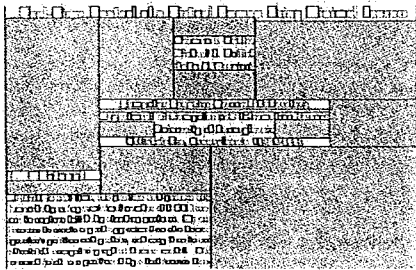


Figure 6: Partial, greedy whitespace cover for a portion of a document, computed using the algorithm described in the paper.

to be bounded by at least some minimum number of connected components on each of its major sides. This essentially eliminates false positive matches and makes the algorithm nearly independent of other parameters (such as preferred aspect ratios).

A gallery of automatically determined whitespace components is shown in Figure 1. The quality function was chosen such that only “tall” whitespace was returned. The resulting whitespace rectangles found by the algorithm correspond exactly to whitespace separating text columns.

4 Non-Axis Aligned Whitespace Rectangles

The algorithms described in the previous section for analyzing page background find axis-aligned whitespace. This section presents an algorithm that finds globally maximal whitespace rectangles on page images at arbitrary orientations. This algorithm eliminates the need for page rotation correction prior to background analysis. That has some advantages for complex documents containing text at multiple orientations in different columns, as well as for significantly degraded documents, for which determination of page rotation (skew) prior to analyzing background structure may be difficult.

The algorithm is resolution independent and takes as input a list of foreground shapes (e.g., character or word bounding boxes or polygons) and a set of parameter ranges; it outputs the N largest non-overlapping maximal whitespace rectangles whose parameters (location, width, height, orientation) fall within the required parameter ranges. It is somewhat analogous to the method described in the previous section, in that it also uses a branch-and-bound algorithm. Details of the algorithm will be described elsewhere [4].

5 Textline Finding

Another essential aspect of document layout is the lines of text. Identifying lines of text reliably is nec-

essary in order to perform OCR. Reliable identification of lines of text also permits the detection of important page layout features such as paragraphs, section headings, etc.

Most past methods to text line finding have either been entirely global (projection methods, Hough transform methods, etc.), or very local (connected component linking, etc.), or they have required a complete page layout analysis as input prior to being able to identify text lines reliably. Global methods have serious problems with multi-column layouts or facing pages in scanned books, in which the orientation or spacing of neighboring text lines may differ significantly (Figure 5 shows two sample instances, plus a successful solution using the approach developed in this section).

We have developed a new approach to textline finding. It combines the advantages of previous approaches without their disadvantages. Like global methods, our approach can take advantage of long-range alignments of textual components, but the method is robust in the presence of multiple columns, like local methods.

The idea is to take the column boundaries identified by the background analysis described in the previous sections (shown in green in Figures 1 and 5) and introduce them as “obstacles” into a statistically robust, least square, globally optimal text line detection algorithm we have previously developed [6]. This match score corresponds to a maximum likelihood match in the presence of Gaussian error on location and in the presence of a uniform background of noise features, as shown in the literature [10].

Perhaps surprisingly, incorporating obstacles into the branch-and-bound textline finding algorithm is simple and does not noticeably increase the complexity of the algorithm on problems usually encountered in practice. This is because of a computational trick we have developed that greatly reduces the dimensionality of the parameter space that needs to be searched. Details can be found in [3]. An evaluation on the UW3 database shows nearly perfect detection of text lines. When used for page skew detection and correction, the method finds estimates of page skews that are within the variability of line orientations within a single page (less than 0.2 degrees on the UW3 database).

6 Reading Order by Topological Sort

As we noted above, recovery of reading order is a hard problem and can depend not only on the geometric layout of a document, but also on linguistic and semantic content. The key idea behind our approach is to determine all the pairwise constraints on reading order that we can, from the geometric

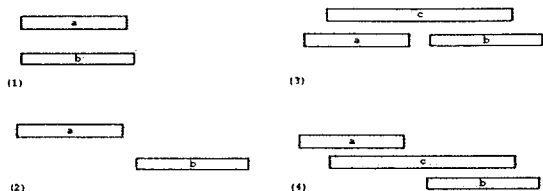


Figure 7: Figure illustrating the partial order criteria. In Example (1), segment *a* comes before segment *b* by criterion one. In Example (2), segment *a* does not come before *b* by criterion one because their *x*-ranges do not overlap (but criterion two may apply). In Example (3), segment *a* comes before segment *b* by criterion two: *a* is completely to the left of *b*, and there is no intervening line segment *c*. In Example (4), segment *a* does not come before *b* by criterion two because segment *c* separates them.

arrangement of text line segments on the page, as well as possibly linguistic relations. This partial order is then extended to a total order of all elements using a topological sorting algorithm [8].

Applying only two ordering criteria turns out to be sufficient to define partial orders suitable for determining reading order in a wide variety of documents:

1. Line segment *a* comes before line segment *b* if their ranges of *x*-coordinates overlap and if line segment *a* is above line segment *b* on the page.
2. Line segment *a* comes before line segment *b* if *a* is entirely to the left of *b* and if there does not exist a line segment *c* whose *y*-coordinates are between *a* and *b* and whose range of *x*-coordinates overlaps both *a* and *b*.

These criteria are illustrated in Figure 7. The first criterion basically ensures that line segments are ordered within their own column. The second criterion orders columns from left to right, but only for columns that fall under a “common heading”.

Examples of recovered reading order using this approach are shown in Figure 3. Note that floating elements, like page headers, footers, and captioned images, were not removed prior to reading order determination. The output of reading order determination therefore contains these floating elements somewhere, usually close to where element is logically referenced. This is acceptable in applications like image-based document reflowing [5]. Otherwise, the floating elements can be removed prior to, or after, reading order determination.

An informal visual inspection of results on documents from the UW3 database suggest that reading order is recovered correctly using this approach in most cases; failures appeared to occur only when the source image contained severely degraded text

areas (e.g., due to poor quality photocopies), or occasionally due to incorrectly determined bounding boxes for images appearing in the text.

7 A Novel Layout Analysis System

With the algorithms described in the previous sections, we can now put together a novel approach to layout analysis, consisting of the following steps:

1. Find tall whitespace rectangles and evaluate them as candidates for gutters, column separators, etc.
2. Find text lines that respect the columnar structure of the document.
3. Identify vertical layout structure (titles, headings, paragraphs) based on the relationship (indentation, size, spacing, etc.) and content (font size and style etc.) of adjacent text lines
4. Determine reading order using both geometric and linguistic information.

To evaluate the performance, the method was applied to the 221 document pages in the “A” and “C” classes of the UW3 database. Among these are 73 pages with multiple columns. The input to the method consisted of word bounding boxes corresponding to the document images. After detection of whitespace rectangles representing the gutters, lines were extracted using the constrained line finding algorithm. The results were then displayed, overlaid with the ground truth, and visually inspected. Inspection showed no segmentation errors on the dataset. That is, no whitespace rectangle returned by the method split any line belonging to the same zone (a line was considered “split” if the whitespace rectangle intersected the baseline of the line), and all lines that were part of separate zones were separated by some whitespace rectangle. Sample segmentations achieved with this method are shown in Figure 1.

8 Scale-Space Layout Analysis

The previous sections have dealt with layout analysis in terms of background whitespace structure, text lines, and reading order. In this section, I briefly describe some work in our lab on scale-space document layout analysis.

Generally, layout analysis methods depend on a number of segmentation parameters, such as the width and height at which whitespace is considered sufficiently salient in order to be considered a layout component. The core idea behind scale space layout analysis is to find a representation of the document layout that makes it possible to perform operations

like matching or retrieval across all scales simultaneously, and to generate segmentations with specific segmentation parameters very quickly. For details, the reader is referred to the references [3]. Here, we will limit ourselves briefly to the applications of such methods. An example of this is shown in Figure 4(a).

Fast Interactive Segmentations Even with the best automatic layout analysis methods, occasionally, layouts need to be created and/or corrected manually. Scale-space layout analysis permits document layout analysis at interactive rates; that is, a user can use GUI controls to modify layout parameters and instantly watch the response. Many documents can be segmented in this way with a simple sweep of the mouse, selecting the horizontal and vertical segmentation thresholds. This permits interactive layout analysis within a few seconds per page.

Layout Based Retrieval Retrieval of documents from document databases based on their physical or logical layout has been described, for example, by Doermann *et al.* [9]. The idea is to first perform a layout analysis of the documents in the database and the query document and then to compare the layouts for the purposes of retrieval. A common problem in layout-based document retrieval occurs when similar documents are segmented slightly differently—the separation of text blocks in one document may fall slightly below or above the segmentation thresholds. Such a document may match a query document poorly, even though a slightly different choice of segmentation parameters might produce a nearly perfect match.

Scale-space layout analysis addresses this problem by not representing documents at a single scale. Instead, when a query document is compared against a document in the database, they are matched with all possible segmentation parameters, and the optimal set of segmentation parameters is selected as part of the matching process. Thereby, problems where slight changes in segmentation parameters cause the quality of match to change significantly are eliminated. An example of this is shown in Figure 4(b).

Segmentation by Example. Many tasks that use document layout analysis are performed on large databases containing pages with similar layouts. Examples are legacy conversions of company memos, patent documents, scientific journals, and medical data sheets. The ability to match page layouts more reliably using scale space segmentation makes it possible to perform segmentation by example. That is, a user adjusts the segmentation parameters of a sample document as needed, and the system adjusts the segmentation parameters on novel documents to

match those on the sample document.

9 Conclusions

This summary paper has described a number of novel algorithms and statistical methods for layout analysis. A combination of globally optimal geometric algorithms with sound statistical models and careful engineering has permitted us to create a new generation of flexible page layout analysis algorithms. This combination yields demonstrably highly robust and versatile layout analysis on documents from the University of Washington Database 3 (UW3). It also holds promise for robust layout analysis in languages using non-Latin writing systems.

The system has been used for building electronic book (e-book) and document conversion applications. We are interested in finding commercial or government partners willing to sponsor the evaluation and adaptation of our software and methods to non-Latin writing systems (Arabic, Chinese, minority languages), and information retrieval and intelligence applications.

References

- [1] H. S. Baird. Background structure in document images. In *H. Bunke, P. S. P. Wang, & H. S. Baird (Eds.), Document Image Analysis, World Scientific, Singapore*, pages 17–34, 1994.
- [2] H. S. Baird, S. E. Jones, and S. J. Fortune. Image segmentation by shape-directed covers. In *Proceedings of the Tenth International Conference on Pattern Recognition, Atlantic City, New Jersey*, pages 820–825, 1990.
- [3] T. M. Breuel. Two algorithms for geometric layout analysis. In *Proceedings of the Workshop on Document Analysis Systems, Princeton, NJ, USA, 2002*.
- [4] T. M. Breuel. An algorithm for finding maximal whitespace rectangles at arbitrary orientations. (under review), 2003.
- [5] T. M. Breuel, W. C. Janssen, K. Popat, and H. S. Baird. Paper-to-pda. In *Proceedings of the International Conference on Pattern Recognition (ICPR'02), Quebec City, Quebec, Canada, 2002*.
- [6] T.M. Breuel. Robust least square baseline finding using a branch and bound algorithm. In *Proceedings of the SPIE - The International Society for Optical Engineering*, page (in press), 2002.

- [7] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. Geometric layout analysis techniques for document image understanding: a review. Technical report, IRST, Trento, Italy, 1998.
- [8] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.
- [9] D. Doermann, C. Shin, A. Rosenfeld, H. Kainiskangas, J. Sauvola, and M. Pietikainen. The development of a general framework for intelligent document image retrieval. In *Document Analysis Systems*, pages 605–632, 1996.
- [10] William Wells III. Statistical approaches to feature-based object recognition. *International Journal of Computer Vision*, 21(1/2):63–98, 1997.
- [11] D. Ittner and H. Baird. Language-free layout analysis, 1993.
- [12] K. Kise, A. Sato, and M. Iwata. Segmentation of page images using the area voronoi diagram. *Computer Vision and Image Understanding*, 70(3):370–82, June 1998.

Automated Layout Recognition

Lynn Golebiowski

Booz | Allen | Hamilton
134 National Business Parkway
Annapolis Junction, Md 20701
golebiowski_lynn@bah.com

1.0 Abstract

To develop document image layout classifiers, each document image is represented by a set of labeled polygons corresponding to the pairwise relationships between objects on the page. “Wanted” and “Unwanted” training sets are used to generate a polygon weight based on frequency of occurrence in both sets (term frequency). Unknown documents are scored by comparing polygons to those occurring in the wanted set. A score, weighted by the term frequency for the matching polygons, is computed. Experiments are performed against the NIST Structured Forms Database based on single and multiple layout collections using a variety of training samples.

2.0 Overview

The intent of this work is to develop techniques for using page layout information to perform automatic page layout identification and, eventually, layout based document image clustering.

In order to provide robustness, several design goals and constraints were applied. First, any approach considered had to permit a high degree of automation. Specifically, the only information to be provided to the algorithm was a sampling of the layout(s) of interest (the “wanted” class) and a corresponding sampling of images that are not of interest (the “unwanted” class). While automated image segmentation techniques are used, no manual or form specific identification of regions of interest are used.

Other design considerations included tolerance to errors in underlying segmentation techniques, image degradation, and page rotation, as well as script independence. A major design goal was to describe the document page layout in such a way as to allow for the application of further analytic techniques to facilitate image retrieval and clustering. Finally, some type of a likelihood metric was desirable.

3.0 Technical Approach

A document image is represented by a set of labeled polygons. These polygons describe the pairwise relationship between objects on the page. By using object arrangement, as opposed to absolute page location, the effects of page translation and rotation can be minimized. In this discussion, the use of only four-sided polygon is presented, but the approach can be easily modified to use polygons with more sides. Once the polygons for an image have been calculated they are compared to the polygons observed across wanted and unwanted training sets and a score is assigned. Image preprocessing, feature calculation, algorithm training, and scoring are presented in the following subsections.

3.1 Image Preprocessing

The skew and the text direction (vertical vs. horizontal) are determined using several unpublished techniques. These methods are based on peak energy projections of connected components computed at varying angles. A rotated and de-skewed version of the image is then obtained. The image is rotated to force the text flow to be horizontal. No attempt is made to ensure the image is upright. For example an image rotated by 180 degrees is not corrected. All subsequent processing is on this oriented image.

3.2 Page Segmentation and Features Calculation

In applying this technique, the objects and features chosen include page level script identification and text lines. Page level script identification was performed using a variant of the Hochberg template matching approach [2]. Text line segmentation was performed using an unpublished script-independent algorithm based on connected component proximities. To improve performance, only the 50 longest lines whose average connected component height was greater than seven pixels were used.

A generic data representation is defined for each feature or object computed for an image. It includes:

- Feature/object type
- Two location points (x/y coordinates) used to represent the object
- An integer qualifier (specific to the object type)
- A string qualifier (specific to the object type)

The location points, integer qualifier, and string qualifier listed above are specific to the feature/object type and are used during the training and scoring process to perform type specific matching. Page level features that do not have a page location are treated as a special case. Table 1 summarizes the information used in applying this approach to the NIST dataset.

Table 1 Object and Feature Information Used

| Object Type | Locators | Integer Qualifier | String Qualifier |
|----------------------------------|------------------------|-------------------|---------------------------------|
| textLine | Line segment end-point | Line Height | Unused |
| Page level script identification | (-1,-1) (-1, -1) | Unused | Text string representing script |

3.3 Training Process

Each image object is represented by two points. For example, text lines are represented by the line end-points. Another possibility would be to represent the line by the upper left and lower right bounding box coordinates. In this way each object on the page defines a line segment. Figure 1 illustrates the line segments used to represent the objects on an image.

Figure 1 Object Line Segments

1040 U.S. Individual Income Tax Return **1988**

For the year Jan.-Dec. 31, 1988, or other tax year beginning 1988, ending

Label L Your first name and initial (if joint return, also give spouse's name and initial) Last name
 A **Berry, K. & Lorraine A. Bayle**
 P Person's identification number
 AS7: 00: 3587

Don't fill in!
 Otherwise, please print or type.

M 73 Mason Street
 N City, town or post office, state, and ZIP code
 E Camden, NE 68522
 R

Purposes: Additional Payment
 Reduction (if return, see instructions)

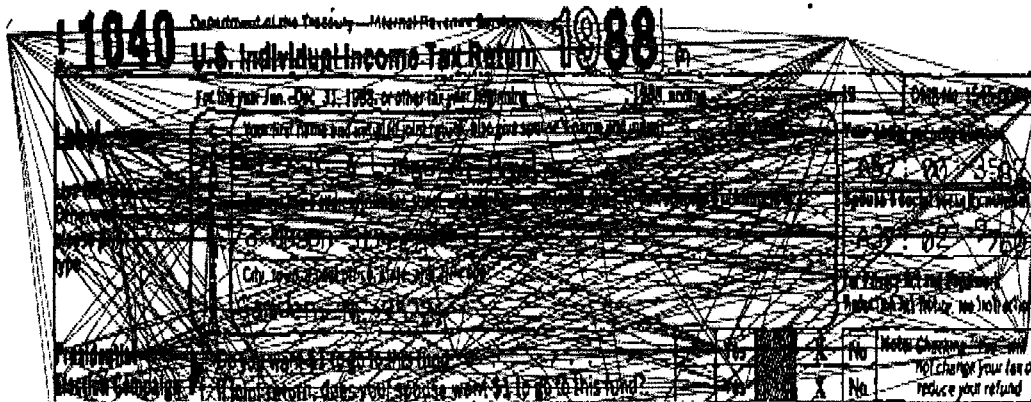
Presidential Election Campaign Do you want \$3 to go to this fund? Yes No
 If joint return, does your spouse want \$3 to go to this fund? Yes No **Not change your tax or reduce your refund**

Filing Status
 1 Single
 2 Married filing jointly (use only one set of income)

A polygon is used to represent pair-wise relationships between objects on the page. It is formed using the four line segments end points representing two objects on the page. An additional tag is used to indicate if line segments forming the polygon crossed. Figure 2 illustrates the polygons formed for a typical image. The following information is used to characterize each polygon:

- Object Type 1
- Object Type 2
- Polygon vertices (currently 4)
- Instances on the page
- Instances across all training documents
- Polygon area (used in matching polygons)
- Crossing flag (indicated if line segments used to form polygon cross)
- Term frequency weight (reflects how good a discriminator the polygon is for its class)
- Turning representation information (used in matching polygons)

Figure 2 ObjectPairs



To train against a specific data set of wanted or unwanted pages, all of the polygons for each image are computed. The resulting polygons are pruned by combining those of like shape and area. Two polygons are said to match if their area is within a threshold (currently ten percent of the larger polygon), their shapes are similar, and the object types and object qualifiers match. A scale, rotation, and translation invariant shape similarity metric [3] is used to determine shape similarity. Matching of object qualifiers is specific to the type of object. In the case of text lines, the integer qualifier contains the line height. In order to match, the height values must be within seven pixels.

At this point, polygons formed for an individual image are combined with the polygons for all other images in the training set. Matching polygons are merged again. The result of training against a set of images is a set of polygons, found in the set of images. The end product is a subset of the polygons found in the training set.

A term-frequency score is used to reflect how indicative a polygon is to a dataset (wanted or unwanted). It is generated by dividing the number of occurrences of the polygon within the dataset by the total number of occurrences across both datasets.

3.4 Scoring Images

An image is scored by comparing polygons formed for an unknown image to the polygons formed across the wanted data set. In many of the datasets used in developing this algorithm it was observed that the term frequency distribution contained a large number concentration of very low numbers. Because polygons with low term frequency contribute little to the final score, the wanted polygon set used for scoring is pruned to include only those polygons with a score above a threshold. This threshold was arbitrarily selected based on the term frequency.

Each polygon from an unknown image is compared with all polygons in the pruned wanted polygon set. The score for the image is incremented by the term frequency weight for the matching polygon. The final score is the sum of all matching scores.

3.5 Polygon Matching Document Clustering

In addition to layout selection, this polygon representation is also being used in an attempt to cluster documents. The potential benefit to layout based clustering is the ability to automatically identify and label frequently occurring images within large collections.

In performing clustering, polygon co-occurrence is used to group documents. A binary feature vector is used as a basis for computing a document similarity matrix. Hierarchical clustering is then performed. Early results are promising, however, further work needs to be done to optimally determine the correct number of clusters in an automated way. Work relating to layout based clustering is documented more fully in [4].

4.0 Evaluation

4.1 NIST Dataset

An evaluation on the NIST forms database was performed. This data set consists of 5590 images spread across 20 different layouts or form faces. All layouts are from the IRS 1040 Package X for 1988. They are distributed as shown in Table 1. The forms have been digitized at 300dpi and the data entered into the forms has been synthesized. The first 500 images were reserved for training and the remaining 5090 images were used for testing.

Table 5 NIST Dataset

| Form Face | Form Number | Training Samples | Testing Samples |
|-----------|-------------|------------------|-----------------|
| 1040_1 | 0 | 83 | 900 |
| 1040_2 | 1 | 82 | 900 |
| 2106_1 | 2 | 9 | 90 |
| 2106_2 | 3 | 8 | 79 |
| 2441 | 4 | 4 | 98 |
| 4562_1 | 5 | 21 | 263 |
| 4562_2 | 6 | 20 | 270 |
| 6251 | 7 | 10 | 97 |
| sch_a | 8 | 44 | 481 |
| sch_b | 9 | 49 | 555 |
| sch_c_1 | 10 | 14 | 198 |
| sch_c_2 | 11 | 12 | 109 |
| sch_d_1 | 12 | 17 | 223 |
| sch_d_2 | 13 | 24 | 242 |
| sch_e_1 | 14 | 29 | 341 |
| sch_e_2 | 15 | 33 | 358 |
| sch_f_1 | 16 | 11 | 86 |
| sch_f_2 | 17 | 7 | 59 |
| sch_se_1 | 18 | 8 | 132 |

Table 5 NIST Dataset

| Form Face | Form Number | Training Samples | Testing Samples |
|-----------|-------------|------------------|-----------------|
| sch_se_2 | 19 | 15 | 109 |

5.1 Experiments Defined

A set of 23 experiments were performed. The first 20 experiments involved targetting a single layout as the wanted class. The unwanted images were those in the other layouts. For example, for the 1040_1 layout there were 83 wanted samples and 417 unwanted samples.

The next experiment involved defining the wanted set to be a collection of layouts. In this case, layouts 0, 2, 4, 6, and 8 were used.

The final two experiments were aimed at examining the lower limit for training samples. In one case, a single sample was used for the wanted set and all of the remaining training images were used for the unwanted set. In the other case, a single target layout sample was used for the wanted class and a single copy of each of the other 19 layouts was used for the unwanted class.

5.2 Results

For each of the experiments where the algorithm was trained on a single layout, the resultant test score distribution showed complete separation between the wanted layout and the other images. Figure 3 shows typical score distributions for some of these experiments. While the score distribution varies from experiment to experiment, there was complete separation between the classes. The actual score distribution did vary for each experiment.

Figure 4 shows the score distribution for the experiment where a collection of forms was defined as the "wanted" class. In this case form numbers 0, 2, 4, 6, and 8 were the wanted class while the remaining were the unwanted class. While there is some overlap in score distribution, the separation is still very clear.

Figure 5 contains the results when a single sample of the wanted class is used for training. The top portion of the figure represents the case where multiple copies of each of the unwanted forms were included in the training set. The bottom portion of the figure contains the results where a single copy of each of the unwanted forms was used.

The results show that, while good separation can be achieved using this technique, the score distributions do not allow for a single threshold value to be selected for all cases. Rather, it is specific to the training set. The desired degree of automation can still be achieved by adding an additional step to the training process. The wanted images are scored and then the mean and standard deviation of the score distribution can be used to automatically select an appropriate threshold. In Figures 3 thru 5, vertical lines are used to show the mean and threshold values corresponding to threshold values one, two, and three standard deviations below the mean.

Training and testing this algorithm against the NIST dataset was performed on machine dual Pentium III 866 MHz processors running RedHat Linux 7.3. When 500 images are used, training on a specific layout took approximately 26 minutes. Scoring images took approximately 2.5 seconds per image. There is room for further improvements by applying further pruning techniques and examining the feature set used.

Figure 3 Sample Single Form Experiment Score Distribution

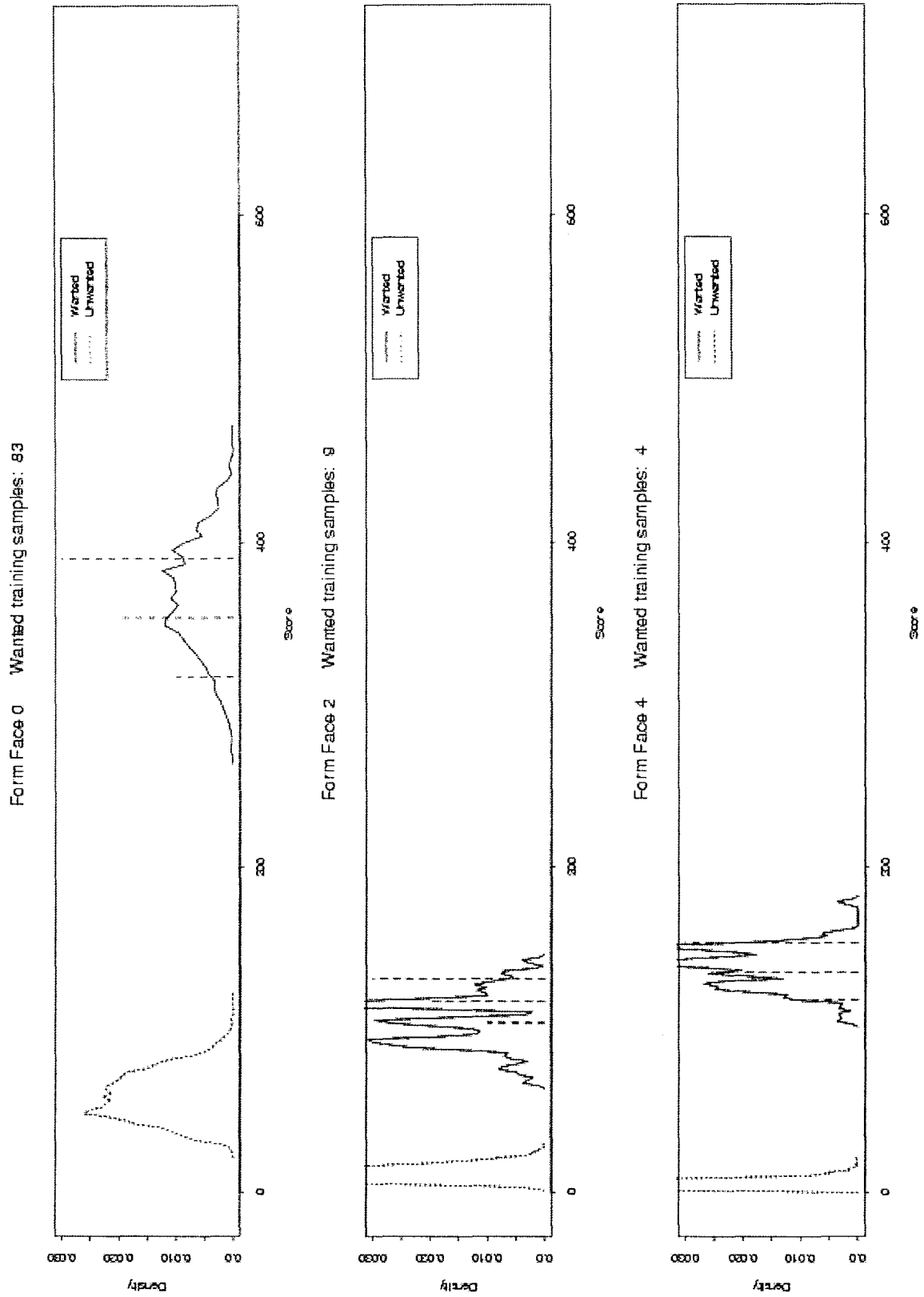
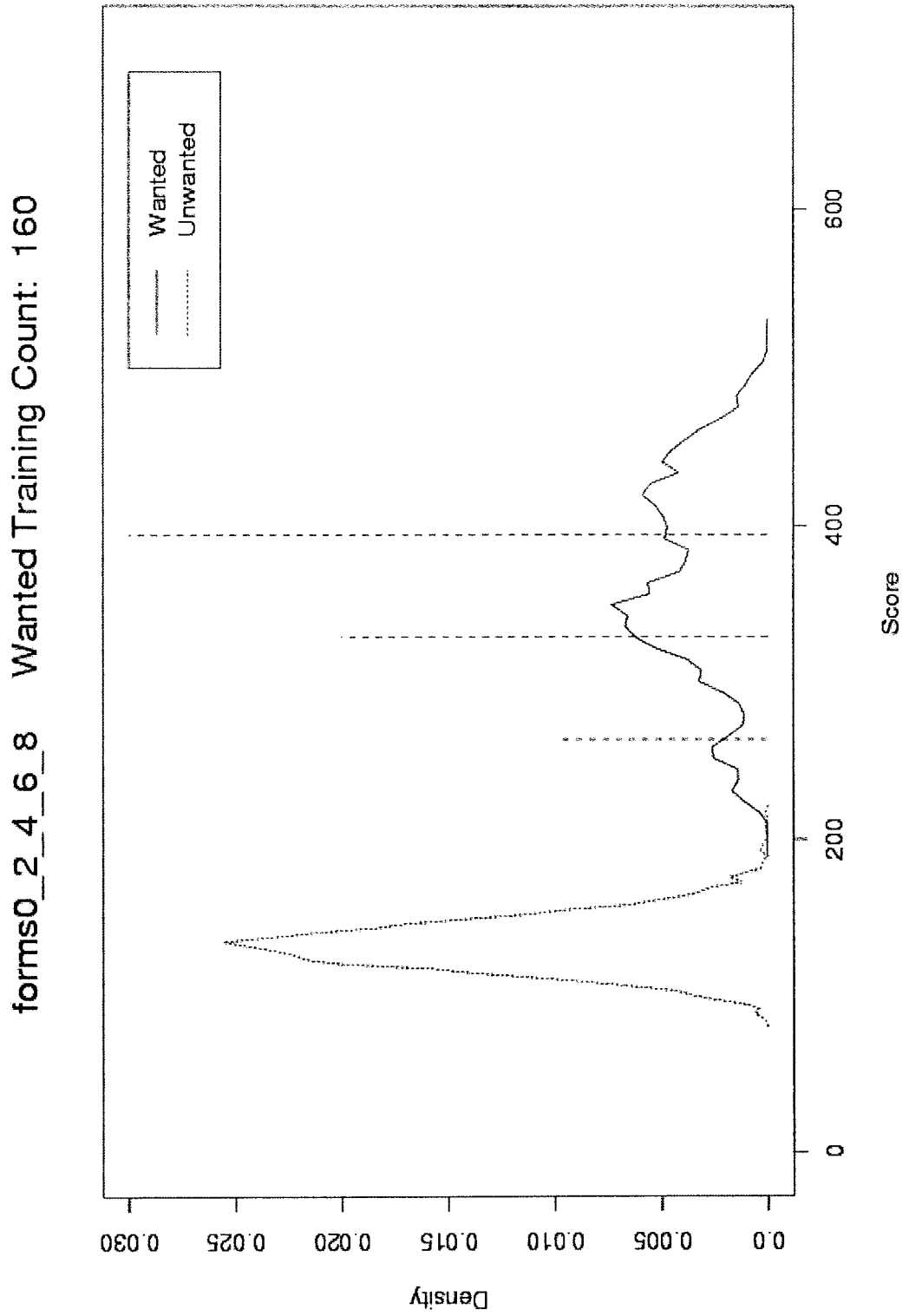
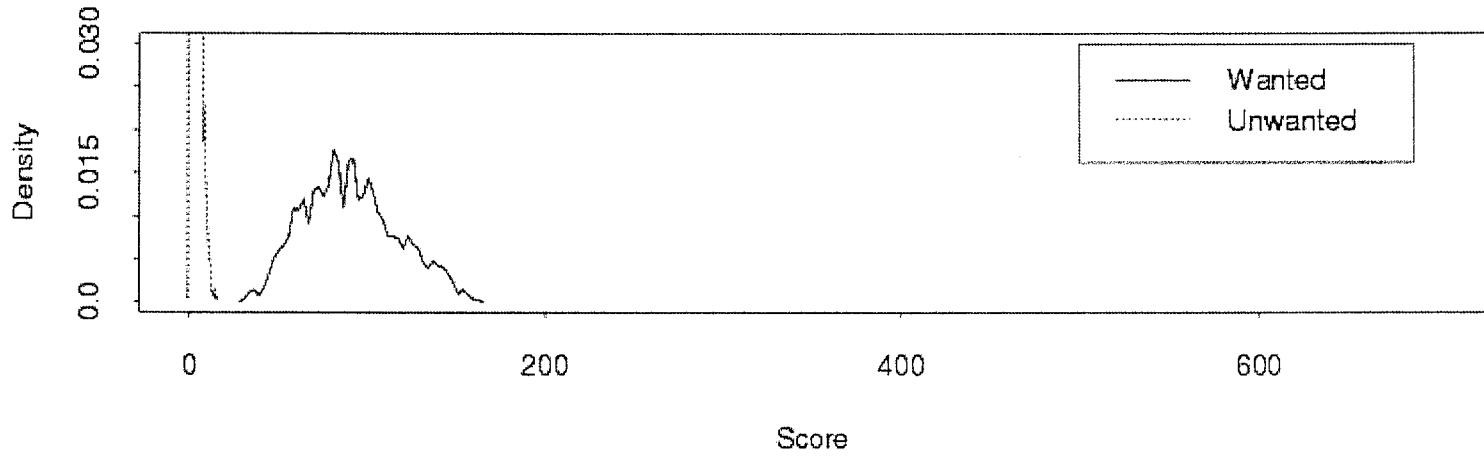


Figure 4 Score Distribution for Multiple Layouts



Form Face 0 Single Wanted Multiple Unwanted



Form Face 0 Single Wanted Single Unwanted

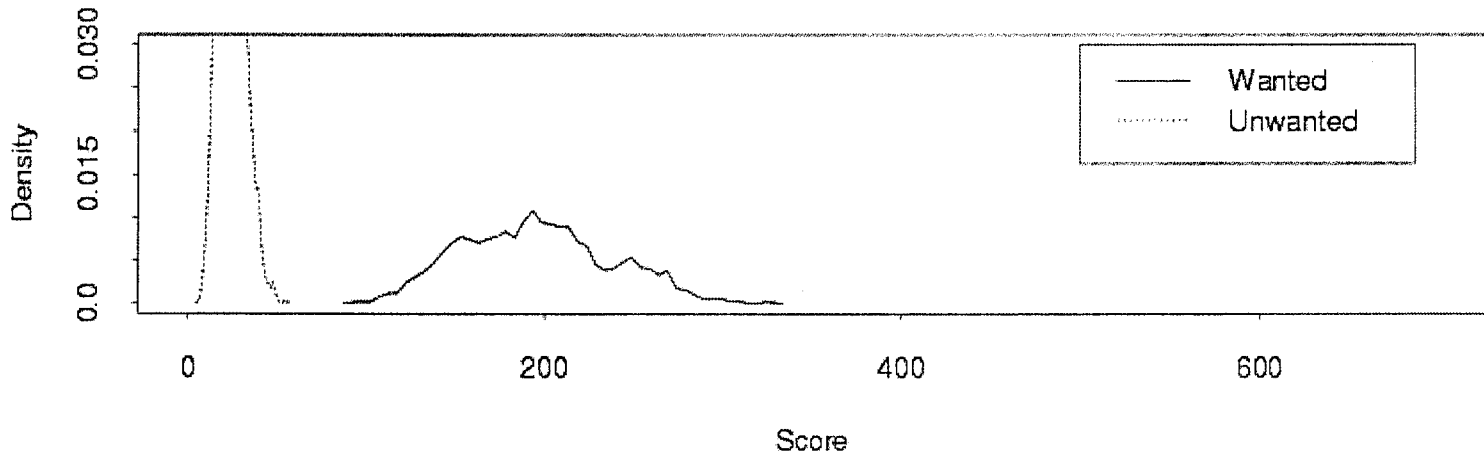


Figure 5 Score Distribution for Limited Training Samples

5.3 Future Plans

Success is being achieved in using a set of labeled polygons to represent document layout. Future plans aimed at improving layout selection include expanding the feature set used, extending the approach to include polygon frequency information, and examining training score distributions to develop a method of automating the selection of a classification threshold. This approach has been demonstrated against the binary document image domain. In the future, this technique will be applied to the color image domain.

Another extension to this work is the attempt to automatically cluster documents based on layout information.

6.0 Acknowledgments

The author is grateful to Alan Sakakihara, Tom Drayer, Dave Doerman, and Ken Cantwell for their feedback on the technical approach presented and their review of this paper.

REFERENCES

- [1] D.L. Dimmick, M.D. Garris, and C.L. Wilson, "NIST Special Database 2 Structured Forms Database", December 1, 1991
- [2] J. Hochberg, P. Kelly, T. Thomas, and L. Kerns, "Automatic Script Identification from Document Images Using Cluster-based Templates", *IEEE Trans. On Pattern Analysis and Machine Intelligence*, Volume 19, 1977
- [3] E. M. Arkin, L.p> Cheew, D.P. Huttenlocher, K. Kedem, and J.L.B Mitchell, "An Efficiently Computable Metric for Comparing Polygonal Shapes", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Volum 13, 1991
- [4] A. Sakakihara, "Clustering Using Normalized Matching Mutual Information for Binary Data, Internal Technical Paper – R522, 2002
- [5] J. Hu, F. Kashi, G. Wilfong, "Document Image Layout Comparison and Classification ", *Proceedings of the Intl. Conf. On Document Analysis and Recognition*, 1999
- [6] R. VeltKamp, "Shape Matching: Similarity Measures and Algorithms", Technical Report UU-CS, 2001
- [7] C. Shin, D. Doerman, "Classification of Document Images Based on Visual Similarity of Layout Structures", *Proceedings SPIE Document Recognition and Retrieval VII*

Layout Based Clustering Using Normalized Matching Mutual Information for Binary Data

Alan "Saki" Sakakihara
Booz | Allen | Hamilton

Abstract

Document images can be represented by binary feature vectors based on page layout. A metric called "normalized matching mutual information (NMMI)" is used to measure the similarity between the feature vectors of each image. Other similarity scores are also studied and evaluated. The NMMI similarity score is implemented with the hierarchical clustering method to group the most similar documents. Next, the F score metric is utilized as a means of determining how well a putative set of clusters matches a truth marked set. After creating a series of clusters over various thresholds, the point at which the F score is maximized is approximated without prior knowledge of the truth marked set.

1 Background

The page layout of a document image may be represented by a set of polygons. These polygons describe the relationships between objects that occur on the page. For example, objects, such as lines of text, would be identified by their endpoints while the polygons resulting from these object pairs might be represented by the quadrilaterals created by connecting the four endpoints [3]. See Figure 1.

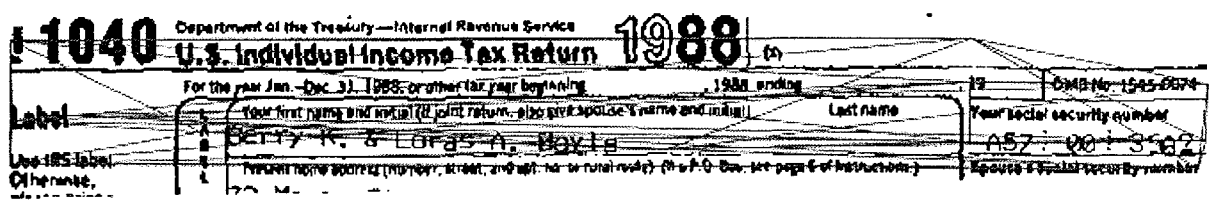


Figure 1: Polygons created by connecting the endpoints of putative lines of text.

Document images are represented by a combination of up to several thousand polygons. Typically, these images are black and white or grayscale text documents.

The goal is to find a robust and accurate method to automatically cluster text document images without prior knowledge of the number of true clusters.

2 Introduction

There are several ways one can cluster data. Clustering output is dependent on how the data is represented and the clustering algorithm chosen. The data, in this case, are the polygons used to represent the relationships between objects occurring in document images.

In this paper, methods to measure document similarity are described. Next, the concept of set relationships is introduced. This is followed by the F score, a metric used to evaluate clustering. Lastly, an algorithm to perform clustering given an unknown number of true clusters is presented along with test results.

3 Normalized Matching Mutual Information (NMMI)

Assume 5 putative text documents are found to contain any combination of up to 30 polygons. The occurrence of the polygons represented in these documents is shown in Table 1.

Table 1: Feature distribution for 5 putative text documents. Each document can be represented by up to 30 polygons.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|
| 1 | Y | Y | Y | Y | Y | Y | Y | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | |
| 2 | Y | Y | Y | Y | N | N | N | Y | Y | Y | Y | Y | Y | N | N | N | N | N | N | N | N | N | N | N | N | Y | Y | Y | N | N | N |
| 3 | N | N | N | N | Y | N | N | N | N | N | N | N | N | Y | Y | Y | Y | N | N | N | N | N | N | N | Y | N | N | N | N | N | |
| 4 | N | N | N | N | N | N | N | N | N | N | N | N | N | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | N | |
| 5 | N | N | N | N | N | Y | Y | Y | Y | Y | Y | Y | Y | N | N | N | N | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | N | N | Y |

Let “no” = 0 and “yes” = 1. The object is to construct a metric that quantifies the similarity between documents based on the occurrence of the polygons in the image.

Define the discrete random variable X to be the occurrence of a polygon. The **entropy** $H(X)$ of X is represented by

$$H(X) = - \sum_{x \in \text{poly } X} p(x) \cdot \log_2 p(x). \quad (1)$$

Put simply, entropy is the numerical measure of the uncertainty of an outcome. Since

$$\lim_{x \rightarrow 0} x \cdot \log_2 x = 0 \quad (2)$$

let

$$0 \cdot \log_2 0 = 0. \quad (3)$$

Entropy values range from zero to infinity. The larger the entropy, the less that is known about the random variable [1]. The entropy of document 1, based on polygon occurrence, is

$$H(\text{document 1}) = -[p(x=0) \cdot \log_2 p(x=0) + p(x=1) \cdot \log_2 p(x=1)] = 0.784.$$

The **pairwise joint entropy** $H(X, Y)$ of discrete random variables X and Y is defined by

$$H(X, Y) = - \sum_{x \in \text{poly } X} \sum_{y \in \text{poly } Y} p(x, y) \cdot \log_2 p(x, y). \quad (4)$$

Pairwise joint entropies also ranges from zero to infinity. The larger the entropy, the less that is known about the co-occurrence of the random variables [1]. The pairwise joint entropy of document 1 and document 2 is

$$\begin{aligned} H(\text{document 1, document 2}) = & -[p(x=0, y=0) \cdot \log_2 p(x=0, y=0) + p(x=0, y=1) \cdot \log_2 p(x=0, y=1) + \\ & p(x=1, y=0) \cdot \log_2 p(x=1, y=0) + p(x=1, y=1) \cdot \log_2 p(x=1, y=1)] = 1.728. \end{aligned}$$

The **conditional entropy** $H(X, Y)$ of discrete random variables X and Y is defined by

$$H(X|Y) = - \sum_{y \in \text{poly } Y} \sum_{x \in \text{poly } X} p(x, y) \cdot \log_2 p(x|y). \quad (5)$$

It also ranges from zero to infinity. Conditional entropy measures the uncertainty of a random variable given knowledge of another random variable. When $p(x|y) = p(x, y) = 0$, it follows that $H(X, Y) = 0$ as a result of (3).

The conditional entropy can be calculated more quickly using the **chain rule of entropy** which, for the pairwise case, states

$$H(X|Y) = H(X, Y) - H(Y) \quad (6)$$

[1].

Mutual information is the reduction in uncertainty of one random variable to due knowledge of another.

$$I(X; Y) = \sum_{x \in \text{poly } X} \sum_{y \in \text{poly } Y} p(x, y) \cdot \log_2 \frac{p(x, y)}{p(x) \cdot p(y)}. \quad (7)$$

Mutual information also ranges from zero to infinity. The greater the mutual information, the more that is known about the co-occurrence of the random variables [1]. Note that mutual information values are symmetric. Hence,

$$I(X; Y) = I(Y; X). \quad (8)$$

Mutual information may be calculated by using the entropy and conditional entropy values

$$I(X;Y) = H(X) - H(X|Y) \quad (9)$$

or, by applying (6), may also be calculated using entropy and pairwise joint entropy values

$$I(X;Y) = H(X) + H(Y) - H(X, Y). \quad (10)$$

This relationship can be most easily be conceived with the help of a Venn diagram as shown in Figure 2 [1].

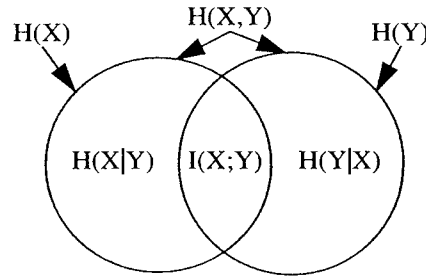


Figure 2: Venn diagram depicting the relationship between entropy and mutual information.

In order to fairly study the relationship between pairs of documents, normalization needs to occur. One would expect large joint entropy values to occur with large mutual information values, and vice versa. Hence, it is logical to divide the mutual information by the pairwise joint entropy. The resulting similarity measure, **normalized mutual information**, is defined in this paper as

$$\frac{I(X;Y)}{H(X, Y)}. \quad (11)$$

Normalized mutual information ranges from 0 to 1 with large scores indicative of a strong relationship. Table 2 lists the normalized mutual information values for each pairwise document combination based on the data in Table 1.

Table 2: Normalized mutual information values for data in Table 1.

| Document A | Document B | $I/H(\text{document A; document B})$ |
|------------|------------|--------------------------------------|
| 1 | 2 | 0.015 |
| 1 | 3 | 0.003 |
| 1 | 4 | 0.217 |
| 1 | 5 | 0.069 |
| 2 | 3 | 0.112 |
| 2 | 4 | 0.165 |
| 2 | 5 | 0.001 |
| 3 | 4 | 0.043 |
| 3 | 5 | 0.113 |
| 4 | 5 | 0.000 |

According to the output in Table 2, documents 1 and 4 exhibit the strongest relationship while documents 4 and 5 display the weakest relationship. Both document pairs are shown in bold. Unfortunately, this measure does not actually provide a measure of co-occurrence. Polygons in documents 1 and 4 never occur together. However, this lack of co-occurrence is in itself a relationship and hence results in a high score.

To consider only the co-occurrence relationships, a new mutual information value must be calculated that is largest when values agree and smallest when they disagree. The result is normalized by the pairwise joint entropy to produce what is defined in this paper as the **normalized matching mutual information** (NMMI), or “nimmi” $\forall \text{nim} - m\bar{e}$, value:

$$M(X;Y) = \frac{\sum_{x \in \text{poly } X} \sum_{y \in \text{poly } Y} (-1)^{x+y} \cdot p(x, y) \cdot \log_2 \frac{p(x, y)}{p(x) \cdot p(y)}}{H(X, Y)}. \quad (12)$$

The normalized matching mutual information value ranges between -1 and 1 with high scores indicative of a strong co-occurrence relationship. The normalized matching mutual information value for documents 1 and 2 is

$M(\text{document 1; document 2}) =$

$$\left(p(x=0, y=0) \cdot \log_2 \frac{p(x=0, y=0)}{p(x=0) \cdot p(y=0)} - p(x=0, y=1) \cdot \log_2 \frac{p(x=0, y=1)}{p(x=0) \cdot p(y=1)} - p(x=1, y=0) \cdot \log_2 \frac{p(x=1, y=0)}{p(x=1) \cdot p(y=0)} + p(x=1, y=1) \cdot \log_2 \frac{p(x=1, y=1)}{p(x=1) \cdot p(y=1)} \right) \cdot \frac{1}{1.728} = 0.134.$$

Note that $H(X, Y) = 1.728$ was calculated previously. The idea is that a higher score should be produced when polygons in both documents exist or both fail to exist. Additionally, differing results should lower the score.

Table 3 lists the normalized matching mutual information values for each pairwise document combination based on the data in Table 1.

Table 3: Normalized matching mutual information values for data in Table 1.

| Document A | Document B | $M(\text{document A; document B})$ |
|------------|------------|------------------------------------|
| 1 | 2 | 0.134 |
| 1 | 3 | -0.050 |
| 1 | 4 | -0.413 |
| 1 | 5 | -0.285 |
| 2 | 3 | -0.251 |
| 2 | 4 | -0.466 |
| 2 | 5 | 0.040 |
| 3 | 4 | 0.200 |
| 3 | 5 | -0.345 |
| 4 | 5 | -0.013 |

According to the output in Table 3, documents 3 and 4 exhibit the strongest relationship while documents 2 and 4 display the weakest relationship. Both document pairs are shown in bold. Considering the frequency of occurrences from each document and their tendency to co-exist, this metric appears reasonable. Polygon matches both appear or both fail to appear in documents 3 and 4 a total of 18 times while such matches occur in documents 2 and 4 only 3 times. One might argue that documents 1 and 2 should have a higher score than documents 3 and 4 since polygons in 1 and 2 both appear or both fail to appear 19 times. However, one must also take into account the frequency of occurrence of each polygon considered. In other words, rare matches will increase the score more than common matches.

4 Set relationships

By setting a NMMI score threshold and selecting the document pairs that exhibit the strongest relationship, one can use NMMI (or any other score that measures pairwise similarity) to perform clustering. It is assumed that document pairs which have a pairwise relationship score above the threshold share a relationship. For example, documents 1, 2, 3, and 4 exhibit a **closed set** relationship when all of the following share a pairwise relationship: (1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4). Thus, n documents have a closed set relationship when each of its documents has a pairwise relationship with each other. Since these relationships are symmetrical, $\binom{n}{2}$ pairwise relationships must exist. Each set of n documents has a **clustering score** of

$$\frac{\text{number of pairwise relationships}}{\binom{n}{2}} \quad (13)$$

This score is dependent on the pairwise similarity scores that determine if two documents share a relationship. The clustering score ranges between 0 and 1 with a higher score indicative of a stronger relationship. A clustering score of 1 implies a closed set relationship.

Documents may also display an **open set** relationship. Suppose only the following documents share a pairwise relationship: (1, 2), (1, 3), (2, 3), (2, 4), (3, 4). Note that documents 1 and 4 do not have a pairwise relationship. This

implies that documents 1, 2, 3, and 4 have an open set relationship. This relationship is defined by the closed set relationship comprised of documents 1, 2, and 3 along with the closed set relationship consisting of documents 2, 3, and 4. 4. The clustering score for open set relationship $\{1, 2, 3, 4\}$ is $\frac{5}{6} = 0.833$. See Figure 3.

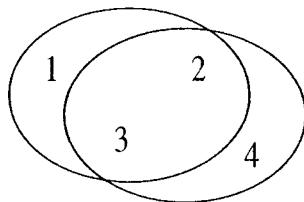


Figure 3: Open set relationship $\{1, 2, 3, 4\}$, closed set relationship $\{1, 2, 3\}$, and closed set relationship $\{2, 3, 4\}$.

If the closed set relationships for a set of documents are disjoint, a **cluster** is formed. See Figure 4.

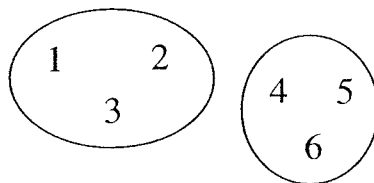


Figure 4: Cluster $\{1, 2, 3\}$, and cluster $\{4, 5, 6\}$.

The NMMI scores generated by the data in Table 3 reflect the similarity of the documents represented by their polygons. A threshold can be set so that only documents whose NMMI score above the threshold are considered to have a pairwise relationship (similar). After determining which documents are similar, clustering scores can be calculated. For a set of n documents, all possible sets of 3, 4, 5, ..., n clusters must be considered. This means

$$\sum_{i=3}^n \binom{n}{i} \tag{14}$$

clustering scores must be calculated.

5 Clustering evaluation

Having defined the NMMI pairwise similarity score, the next task is to use this metric to cluster data. There are a multitude of methods for clustering data but most operate given a pre-determined number of clusters. The object is to perform clustering and stop when the optimal number of clusters is reached. In order to make this determination, a metric for comparing cluster sets must be considered. This is where the **F measure** is used [4].

Suppose the goal is to compare the clusters shown in Figure 5 with the clusters in Figure 6.

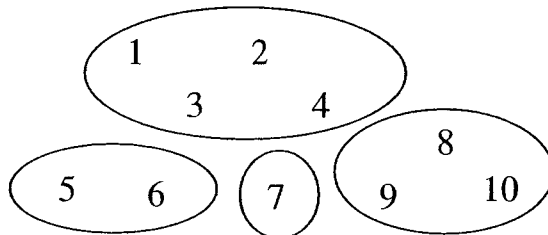


Figure 5: Clusters $\{1, 2, 3, 4\}$, $\{5, 6\}$, $\{7\}$, $\{8, 9, 10\}$.

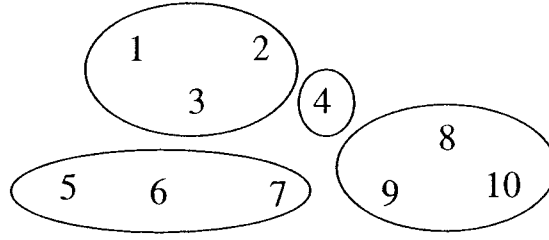


Figure 6: Clusters {1, 2, 3}, {4}, {5, 6, 7}, {8, 9, 10}.

Denote the clusters in Figure 5 by $t_1 : \{1, 2, 3, 4\}$, $t_2 : \{5, 6\}$, $t_3 : \{7\}$, and $t_4 : \{8, 9, 10\}$. Denote the clusters in Figure 6 by $c_1 : \{1, 2, 3\}$, $c_2 : \{4\}$, $c_3 : \{5, 6, 7\}$, and $c_4 : \{8, 9, 10\}$.

The **precision** for clusters t and c is defined as

$$\text{Precision}(t, c) = \frac{n_{tc}}{n_c} \quad (15)$$

where n_{tc} is the number of members of cluster t in cluster c and n_c is the number of members of cluster c . The **recall** for clusters t and c is defined as

$$\text{Recall}(t, c) = \frac{n_{tc}}{n_t} \quad (16)$$

where n_{tc} is the number of members of cluster t in cluster c and n_t is the number of members of cluster t . Both the precision and recall values range between 0 and 1 with higher scores indicative of a closer match. Hence, the non-zero precision and recall values are

$$\begin{aligned} \text{Precision}(t_1, c_1) &= \frac{3}{4} = 0.75 & \text{Recall}(t_1, c_1) &= \frac{3}{3} = 1, \\ \text{Precision}(t_1, c_2) &= \frac{1}{4} = 0.25 & \text{Recall}(t_1, c_2) &= \frac{1}{1} = 1, \\ \text{Precision}(t_2, c_3) &= \frac{2}{2} = 1 & \text{Recall}(t_2, c_3) &= \frac{2}{3} \approx 0.667, \\ \text{Precision}(t_3, c_3) &= \frac{1}{1} = 1 & \text{Recall}(t_3, c_3) &= \frac{1}{3} \approx 0.333, \\ \text{Precision}(t_4, c_4) &= \frac{3}{3} = 1 & \text{Recall}(t_4, c_4) &= \frac{3}{3} = 1. \end{aligned}$$

The **F measure** of cluster t and c is defined as

$$F(t, c) = \frac{2 \cdot \text{Precision}(t, c) \cdot \text{Recall}(t, c)}{\text{Precision}(t, c) + \text{Recall}(t, c)} \text{ if } \text{Precision}(t, c) + \text{Recall}(t, c) \neq 0, \text{ otherwise } F(t, c) = 0. \quad (17)$$

The F measure ranges between 0 and 1 with higher scores indicative of a closer match. Hence, the non-zero F measure values are $F(t_1, c_1) \approx 0.857$, $F(t_1, c_2) = 0.4$, $F(t_2, c_3) = 0.8$, $F(t_3, c_3) = 0.5$, and $F(t_4, c_4) = 1$. The F measures obtained from comparing clusters can be combined to produce a single numeric score indicative of how well a putative set of clusters matches a truth marked set of clusters. Call this value the **F score** and define it as

$$F = \sum_i \frac{|t_i|}{n} \cdot \max\{F(t_i, c_1), F(t_i, c_2), \dots, F(t_i, c_{|c|})\} \quad (18)$$

where $|t_i|$ = the number of members in cluster t_i , $|c|$ = the number of clusters in the set that corresponds with c , and n = the total number of members (documents). The F score ranges between 0 and 1 with higher scores indicative of a closer match. Hence, the F score that corresponds with the data shown in Figures 5 and 6 is

$$F = \frac{4}{10} \cdot 0.857 + \frac{2}{10} \cdot 0.8 + \frac{1}{10} \cdot 0.5 + \frac{3}{10} \cdot 1 = 0.853.$$

[4].

6 Hierarchical clustering

The basic process of **hierarchical clustering** is summarized as follows:

1. Start by assigning each item to its own cluster.
2. Find the most similar pair of clusters and merge them into a single cluster.
3. Measure the similarity between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all items are clustered into a single cluster.

Measuring the similarity between clusters can be done in several ways. In **single-link clustering** (nearest neighbor method), the distance between clusters is the shortest distance from any member of one cluster to any member of another cluster. In **complete-link clustering** (furthest neighbor method), the distance between clusters is the longest distance from any member of one cluster to any member of another cluster. In **average-link clustering** (group average method), the distance between clusters is the average distance from any member of one cluster to any member of another cluster [5]. These are all arithmetic methods while more sophisticated geometric or algebraic algorithms include **centroid**, **median**, and **minimum variance** [2]. The method used for this study is average-link clustering.

Consider an example where 5 documents are provided that produce the following $\binom{5}{2} = 10$ NMMI scores:

$$\begin{array}{lllll} M(1, 2): 0.84 & M(1, 3): 0.67 & M(1, 4): 0.34 & M(1, 5): 0.36 & M(2, 3): 0.98, \\ M(2, 4): 0.14 & M(2, 5): 0.25 & M(3, 4): 0.28 & M(3, 5): 0.19 & M(4, 5): 0.69. \end{array}$$

In the first iteration of hierarchical clustering, each document forms its own cluster. Hence, the clusters are {1}, {2}, {3}, {4}, {5}. By merging the 2 most similar documents into a single cluster, the new set of clusters is {1}, {2, 3}, {4}, {5}. A threshold of 0.98 is associated with this set of clusters. The next iteration yields {1, 2, 3}, {4}, {5} with a threshold of $M(\{1, 2, 3\}, 4) = \frac{M(1, 2) + M(1, 3)}{2} = 0.755$. The iteration after this produces {1, 2, 3}, {4, 5} with a threshold of $M(4, 5) = 0.69$. Finally, all the data is merged into a single cluster {1, 2, 3, 4, 5} having a threshold of $M(\{1, 2, 3\}, \{4, 5\}) = \frac{M(1, 4) + M(1, 5) + M(2, 4) + M(2, 5) + M(3, 4) + M(3, 5)}{6} = 0.26$.

The question is “at what point is the cluster optimal?” If the data is truth marked, then the true clusters will be known, and an F score can be obtained to measure how well the putative set of clusters matches the actual set of clusters. Suppose the truth marked set reveals {1, 2}, {3, 4, 5} as the correct cluster. This being the case, the putative set of clusters, their thresholds, deltas, and F scores are shown in Table 4.

Table 4: Data for example with truth clusters {1, 2}, {3, 4, 5}.

| Putative cluster | Pairwise similarity score threshold | Score threshold delta | F score |
|-------------------------|-------------------------------------|-----------------------|------------|
| {1}, {2}, {3}, {4}, {5} | NULL | NULL | 0.567 |
| {1}, {2, 3}, {4}, {5} | 0.98 | NULL | 0.567 |
| {1, 2, 3}, {4}, {5} | 0.755 | 0.225 | 0.62 |
| {1, 2, 3}, {4, 5} | 0.69 | 0.065 | 0.8 |
| {1, 2, 3, 4, 5} | 0.26 | 0.43 | 0.678 |

Ideally, the highest F score (indicating most correct cluster) will correspond with a peak in the pairwise similarity score threshold delta. That is, given clean data, the highest score threshold delta should occur just after the cluster receiving the highest F score.

7 Test results

To test the theory that the highest score threshold delta will occur just after the cluster receiving the highest F score, 6 independent tests were run on disjoint sets of 100 National Institute of Standards and Technology (NIST) Group 4 compressed binary Tagged Image File Format (tiff) document images from the Structured Forms Database. Each of these images belong to one of up to 20 possible document formats in the IRA 1040 Package X for 1998. The overall quality of these images is very good. In conducting these tests, no information was assumed as to the actual number of formats in each test set.

NMMI scores and average-link clustering was performed on the test set. The second of the size independent tests using 100 NIST images produced output represented by the **dendrogram** in Figure 7. Reading from the bottom up, this dendrogram depicts the sequence of successive fusions that occur among the various documents as the NMMI score increases [2]. Each branch in the dendrogram is associated with a value formatted as

<document number>_<true cluster number>

Note how branches of the same truth cluster tend to fuse. The dotted line indicates the point at which the putative cluster set matches the true cluster set perfectly (F score=1.0). It corresponds with an NMMI threshold of 0.109.

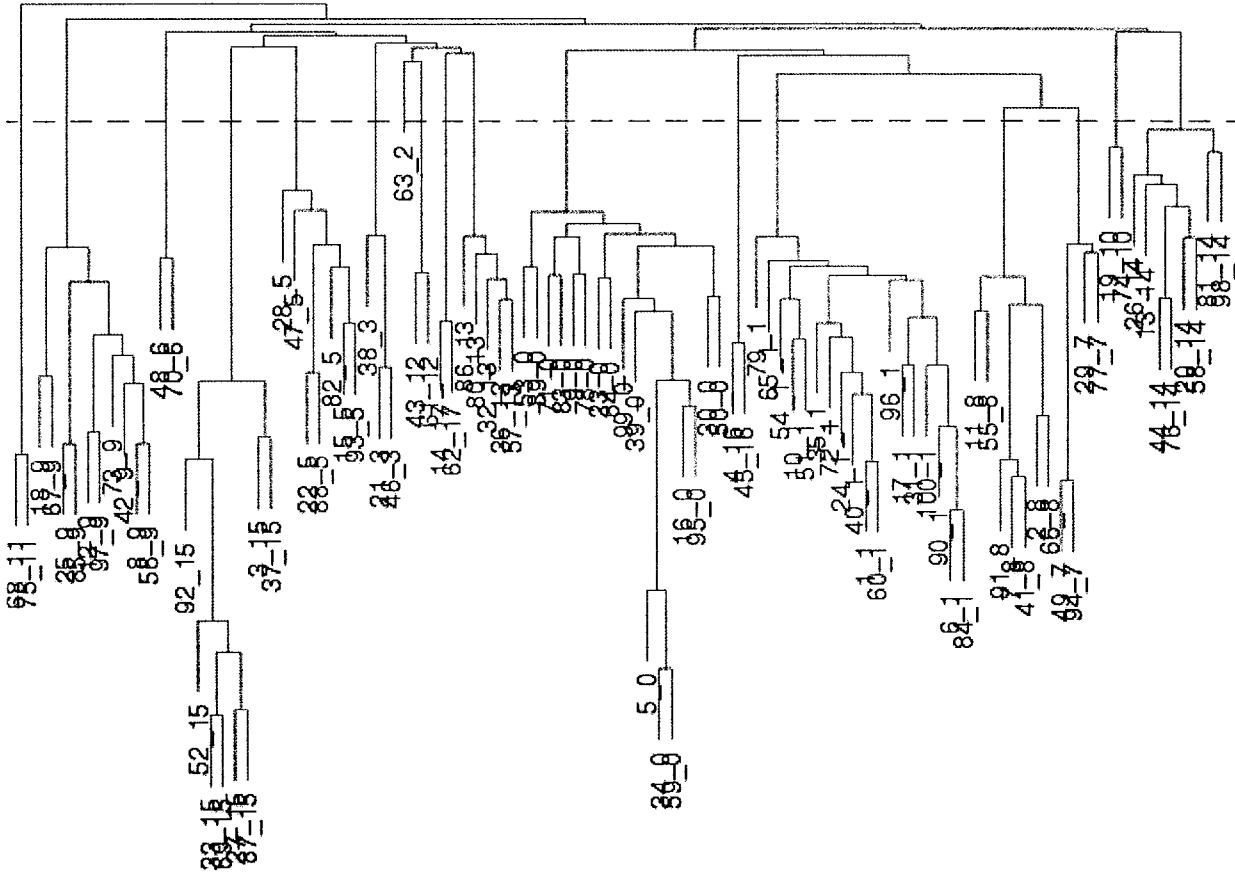


Figure 7: Dendrogram based on NMMI scores and average-link clustering using Test 2 images from NIST Structured Forms Database.

Each test produced 98 threshold deltas. According to the stated theory, if the highest score threshold is produced on the same output line as the putative set comprised of c clusters, then the highest F score should be found for the set comprised of $c+1$ clusters (as in Table 4). For Test 1, the 5 highest score threshold deltas were found to indicate the largest F scores for clusters numbering 99, 18, 19, 20, and 81, respectively. In actuality, the highest F score was 1.0, a perfect match, found with 19 clusters. A summary of the 5 largest threshold deltas for all 6 tests is displayed in Table 5. For simplicity, let the n th highest score threshold delta be denoted by Δ_n , its corresponding F score represented by F_n , and the number of clusters corresponding to this F score denoted by C_n . Note that for these tests, m is the number of documents, $n \in \{1, 2, 3, \dots, m-2\}$, and $C_n \in \{2, 3, 4, \dots, m-1\}$. The bold values depict the clusters receiving the highest F score for each test in the table. Note in the “Maximum $F_n : C_n$ ” column that hierarchical clustering for Test 3 did not result in a perfect match with the truth set for any threshold. This column shows the best results.

Table 5: Summary of 6 independent tests to find optimal clusters using disjoint sets of 100 NIST Structured Forms Database images.

| Test | Maximum $F_n : C_n$ | $\Delta_1 : F_1 : C_1$ | $\Delta_2 : F_2 : C_2$ | $\Delta_3 : F_3 : C_3$ | $\Delta_4 : F_4 : C_4$ | $\Delta_5 : F_5 : C_5$ |
|--------|---------------------|------------------------|------------------------|------------------------|-------------------------|------------------------|
| Test 1 | 1.0: 19 | 0.087: 0.312: 99 | 0.036: 0.978: 18 | 0.029: 1.0: 19 | 0.017: 0.994: 20 | 0.014: 0.496: 81 |
| Test 2 | 1.0: 17 | 0.065: 0.340: 94 | 0.034: 0.325: 95 | 0.032: 0.279: 98 | 0.025: 0.966: 16 | 0.024: 0.357: 93 |
| Test 3 | 0.988: 16 | 0.097: 0.261: 99 | 0.036: 0.276: 98 | 0.028: 0.290: 96 | 0.027: 0.975: 14 | 0.027: 0.400: 86 |
| Test 4 | 1.0: 18 | 0.051: 1.0: 18 | 0.019: 0.316: 97 | 0.018: 0.431: 88 | 0.017: 0.328: 96 | 0.015: 0.173: 2 |
| Test 5 | 1.0: 19 | 0.099: 0.305: 99 | 0.039: 0.347: 96 | 0.025: 0.331: 97 | 0.025: 0.989: 21 | 0.024: 0.395: 90 |
| Test 6 | 1.0: 18 | 0.065: 0.293: 99 | 0.051: 0.995: 19 | 0.023: 1.0: 18 | 0.019: 0.351: 94 | 0.017: 0.168: 2 |

Clearly, there is a tendency for numerous clusters to yield a large threshold delta. However, by overlooking the most numerous cluster values, the highest score threshold delta becomes a fairly good indicator of the cluster set that yields the maximum F score. In fact, for the given data set, this method is never off by more than 2 clusters.

Up to now, the goal has been to have a threshold independent method of clustering data into an undetermined number of sets. Knowing that the values associated with the most numerous clusters might tend to skew things a bit, consider the $\Delta_n : F_n : C_n$ values such that $C_n \in \{2, 3, 4, \dots, [0.9m]\}$ where $[x]$ is the greatest integer function of x ; the greatest integer less than x . One might call 0.9 a threshold but for now, consider it a fixed value that only throws out 10-11% of the possible outcomes. If the user believes there is a good chance that the actual number of clusters is near m , then this method should simply not be used. Table 6 illustrates the $\Delta_n : F_n : C_n$ values given the above restriction. Again, the bold values depict the clusters receiving the highest F score for each test in the table. With this output, the C_n producing the maximum F_n is always in the top five Δ_n and the Δ_1 value always corresponds to an F score of at least 0.966, indicating a very close match.

Table 6: Summary of 6 independent tests to find optimal clusters using disjoint sets of 100 NIST Structured Forms Database images. Clusters limited to 89 or less.

| Test | Maximum $F_n : C_n$ | $\Delta_1 : F_1 : C_1$ | $\Delta_2 : F_2 : C_2$ | $\Delta_3 : F_3 : C_3$ | $\Delta_4 : F_4 : C_4$ | $\Delta_5 : F_5 : C_5$ |
|--------|---------------------|------------------------|------------------------|-------------------------|------------------------|------------------------|
| Test 1 | 1.0: 19 | 0.036: 0.978: 18 | 0.029: 1.0: 19 | 0.017: 0.994: 20 | 0.014: 0.496: 81 | 0.012: 0.465: 84 |
| Test 2 | 1.0: 17 | 0.025: 0.966: 16 | 0.016: 0.201: 2 | 0.015: 1.0: 17 | 0.014: 0.955: 24 | 0.014: 0.989: 18 |
| Test 3 | 0.988: 16 | 0.027: 0.975: 14 | 0.027: 0.400: 86 | 0.023: 0.988: 16 | 0.019: 0.393: 87 | 0.017: 0.913: 25 |
| Test 4 | 1.0: 18 | 0.051: 1.0: 18 | 0.018: 0.431: 88 | 0.015: 0.173: 2 | 0.014: 0.983: 20 | 0.013: 0.852: 40 |
| Test 5 | 1.0: 19 | 0.025: 0.989: 21 | 0.023: 0.181: 2 | 0.022: 0.978: 18 | 0.018: 0.995: 20 | 0.013: 1.0: 19 |
| Test 6 | 1.0: 18 | 0.051: 0.995: 19 | 0.023: 1.0: 18 | 0.017: 0.168: 2 | 0.017: 0.445: 83 | 0.016: 0.396: 88 |

8 Conclusion

Using an information theoretic approach, the similarity relationships between documents represented by a set of polygons can be measured. The similarity metric used for this application is called normalized matching mutual information (NMMI). Given the number of clusters desired, NMMI is easily employed with hierarchical clustering to produce the optimal cluster sets. The challenge is using NMMI similarity scores to perform clustering for an unspecified number of clusters. Using clean data, there is a tendency for a large change in the NMMI score clustering threshold to indicate a good match between the putative cluster set and the true cluster set, especially when the putative cluster set is programmed to ignore the most numerous clusters.

Acknowledgments

The author would like to thank Lynn Golebiowski of Booz | Allen | Hamilton for plotting the dendrogram and providing the polygon data required to represent the various image documents described in this paper.

References

- [1] T. M. Cover and T. A. Joy. *Elements of Information Theory*. John Wiley & Sons, Inc., New York, 1991, 12-22.
- [2] W. J. Krzanowski. *Principles of Multivariate Analysis*. Oxford Science Publications, 1988, 89-94.
- [3] L. M. Golebiowski. "Automated Layout Recognition." Proceedings Symposium on Document Image Understanding Technology (SDIUT) 2003, Department of Defense, 2002.
- [4] M. Steinbach, G. Karypis, and V. Kumar. "A Comparison of Document Clustering Techniques (Technical Report #00-034)." Department of Computer Science and Engineering, University of Minnesota.
- [5] S. P. Borgatti. "How to Explain Hierarchical Clustering." University of South Carolina. 1994.

Automatic Forms Processing in the NIST forms database Document Image Understanding Technology 2003

Carson Cumbee
U.S. Department of Defense

Abstract

A method is introduced to automatically align document images of static forms, and to extract regions of interest. The method described takes a set of examples of a given form, and finds the pair of connected components that through a rigid rotation best aligns the sample forms. After this set of components is identified, all of the remaining similar forms are found automatically and aligned to form a template. Regions of interest are found examining the properties of the template. This method assumes that forms are scanned in at the same resolution but is entirely rotationally invariant.

1.) Introduction

Forms are a subset of document images that are difficult for general purpose OCR systems to process because of the unpredictable manner in which characters, lines, and symbols are laid out on a page. Because of the highly variable nature of forms, it is useful to construct a system that will learn the structure of an unknown form just by analyzing a few samples of it. There are several software packages that exist for the automatic processing of forms. Mitek, Captiva, and ReadSoft are some of the companies with automatic form processing products[1][2][3].

In 1991 NIST created a dataset of forms scanned in at 300dpi with synthetic entries to supplement research in static form processing. The dataset has 20 different tax forms and 5590 total images.

2.) Form Representation by Connected Components

Figure 1 and 2 shows a portion of a typical form in the NIST dataset.

Schedule D (Form 1040) 1988
Attachment Sequence No. 12 Page 2
Name(s) as shown on Form 1040. (Do not enter name and social security number if shown on other side.) Your social security number
A19 80 7180
Rewan C. & Post O. Dale
Part III Summary of Parts I and II
18 Combine lines 8 and 17, and enter the net gain or (loss) here. If result is a gain, also enter the gain on Form 1040, line 13. 18 229 -
19 If line 18 is a (loss), enter here and as a (loss) on Form 1040, line 13, the smaller of:
a The (loss) on line 18; or
b (\$3,000) or, if married filing a separate return, (\$1,500). 19 (0 -)
Note: When figuring which amount is smaller, treat them as if they were positive numbers.
Summary From 1988 to 1989

Figure 1: A partial image from the NIST forms database

Schedule D (Form 1040) 1988 Attachment Sequence No. 12 Page 2

Name(s) as shown on Form 1040. (Do not enter name and social security number if shown on other side.)

Hampton Z. and Skidmore P. Taylor Your social security number
A 99 : 26 : 9320

Part III Summary of Parts I and II

18 Combine lines 8 and 17, and enter the net gain or (loss) here. If result is a gain, also enter the gain on Form 1040, line 13. 18 (0)

19 If line 18 is a (loss), enter here and as a (loss) on Form 1040, line 13, the smaller of:

a The (loss) on line 18; or

b (\$3,000) or, if married filing a separate return, (\$1,500). 19 (0)

Note: When figuring which amount is smaller, treat them as if they were positive numbers.

Part IV Computation of Capital Loss Carryovers From 1988 to 1989
(Complete this part if the loss on line 18 is more than the loss on line 19.)

Figure 2: Another instance of the form in Figure 1

The approach outlined in this paper begins by considering the document to be a set of connected components (defined as a single continuous black region). Figure 3 shows the document in Figure 2 as a collection of X's where the center of connected components are located. Only components that have an area of 100 pixels or greater are shown.

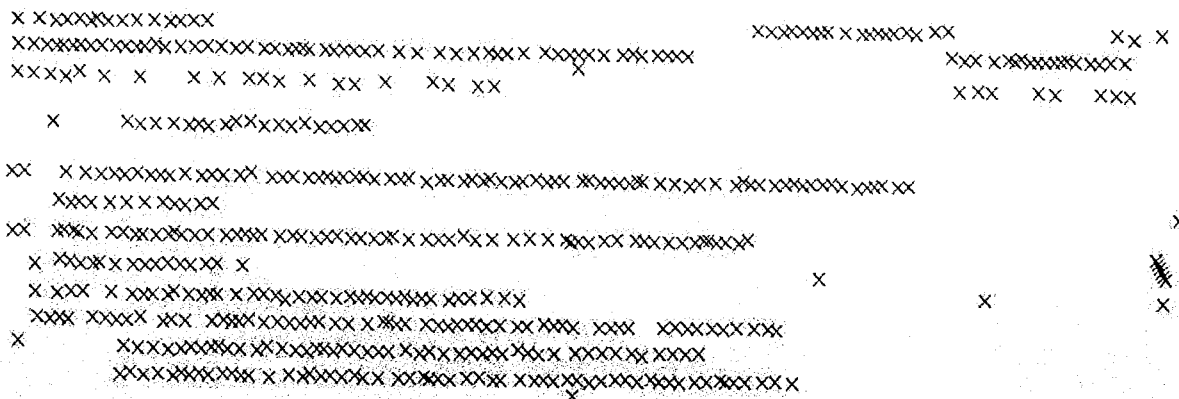


Figure 3: Figure 2 as a collection of points that represent connected components

Figure 3 shows how the layout of a form can be encapsulated by the positions of connected components on a page. Connected components are easy to extract from binary documents, and resistant to being corrupted by a moderate amount noise so algorithms based on them should have few points of failure.

A pair of images can be aligned if the same pair of connected components on each page is identified. The algorithm for alignment described in the next section assumes that the images of forms are acquired through the same scanning process (same dpi) so the most apparent cause of noise is skew and orientation. Even so a single connected component can vary quite a bit from page to page. Once the same pair of connected components are found on all of the samples, a simple rotation and translation can line a pair of images against each other. Because document images are acquired through a real world process, there are variations from document to docu-

ment. It is useful to consider the optimal pair of connected components because the variability of scanned forms prevents many pairs of connected components from being good candidates. It is often the case that forms may only have a small region that is similar across all forms.

3) Automatic Alignment

Given a set of N examples of a certain form, find the pair of connected components over a certain size on each example that when rotated and translated will best align all N examples.

Step 1) Extract all connected components from each image, discarding components under a size A.

Step 2) Find all pairs of connected components in each image and the distance between each pair.

Step 3) For each image, sort the pairs from longest distance to shortest.

Step 4) Starting with the first image's sorted list of pairs, find pairs in the other images with a similar distance within a given tolerance ϕ (by looping through the sorted list).

Step 5) For each pair in the other images, rotate and translate the pair to line up with the first image's pair and rotate and translate all of the other connected components.

Step 6) Count how many connected components are within a distance σ of the first image's components. If this number is less than the maximum of the minimum number of previous matches move to next shortest line in the first document and repeat 4-6 till completion. If the minimum number of matches is higher than the current number of matches, then replace the current maximum of minimum number of matches and continue 4-6.

After this loop is complete a pair of components on each the sample images is selected. This set of pairs are the pairs that have the highest minimum number of matches for all of the sample documents. The results of this approach for Figure 1 and Figure 2 are the components highlighted in Figure 4.

Schedule D (Form 1040) 1988
 Attachment Sequence No. 12
 Name(s) as shown on Form 1040. (Do not enter name and social security number if shown on other side.)
 Your social security
 A19: 80:

Schedule D (Form 1040) 1988
 Attachment Sequence No. 12
 Name(s) as shown on Form 1040. (Do not enter name and social security number if shown on other side.)
 Your social security
 Hampton 7. and Skidmore D. M...
 A19: 80:

Figure 4: The "Sc" component and the "c" component are selected via the above algorithm

| Schedule D (Form 1040) 1988 | | Attachment Sequence No. 12 | Page 2 |
|--|--|-----------------------------|--------|
| Name(s) as shown on Form 1040. (Do not enter name and social security number if shown on other side.) | | Your social security number | |
| Hampton Z. and Skidmore P. Taylor Rowan C. and Peter C. Day | | 899 : 86 : 9320 | |
| Part III Summary of Parts I and II | | | |
| 18 | Combine lines 8 and 17, and enter the net gain or (loss) here. If result is a gain, also enter the gain on Form 1040, line 13. | 18 | (00 -) |
| 19 | If line 18 is a (loss), enter here and as a (loss) on Form 1040, line 13, the smaller of: | | |
| a | The (loss) on line 18; or | | |
| b | (\$3,000) or, if married filing a separate return, (\$1,500). | 19 | (00 -) |
| Note: When figuring which amount is smaller, treat them as if they were positive numbers. | | | |
| Part IV Computation of Capital Loss Carryovers From 1988 to 1989 (Complete this part if the loss on line 18 is more than the loss on line 19.) | | | |

Figure 5: Figure 1 and 2 aligned by the above algorithm

4) Form retrieval

Once the optimal pair of components are selected, they can in association with the remaining components be used to retrieve similar forms. The process is simple, just take the unknown image and extract the connected components that are larger than the size A, find all pairs of components that are the same distance from one another as the optimal pair (within a tolerance phi). Rotate and translate this pair to line up with the optimal pair, and count how many components line up with the training components. Using this method, one can create a ranked list of images in an unknown set where the highest scoring images are the ones most likely to correspond to the sample forms. The NIST database has 20 forms with 5590 total images. To illustrate this method of form retrieval , 10 samples of form "Schedule D page 2" were used as examples, and the remaining 5580 files were scored by matching components to the optimal pair found in these samples. All of the remaining 232 "Schedule D page 2" were ranked higher than all of the remaining 19 documents, demonstrating perfect image retrieval. Figure 6 is a curve of the scores for all 5580 test images. The images that are "Schedual D page 2" are the high scoring ones on the right.

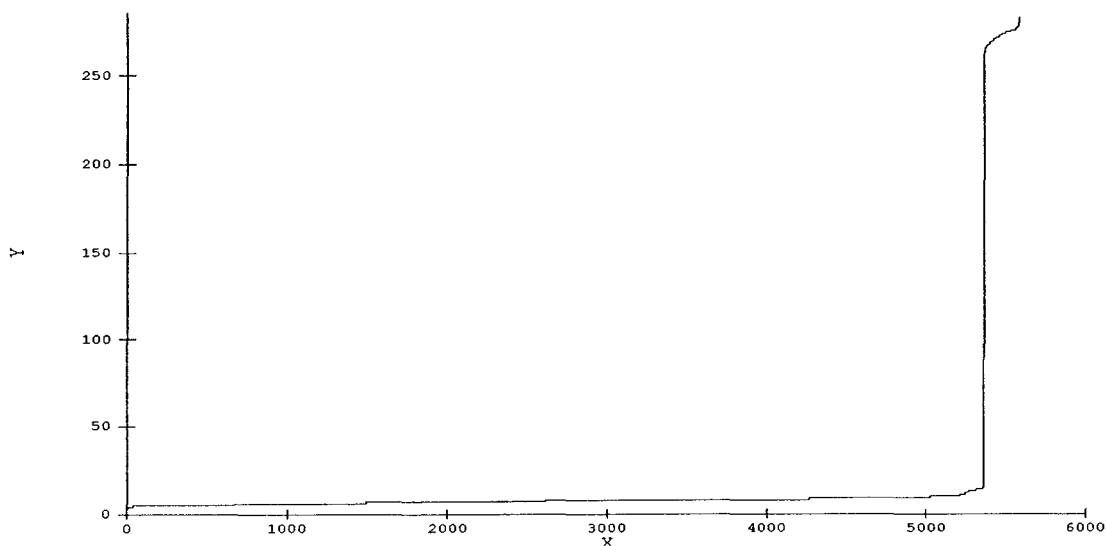


Figure 6: Scores for retrieving "Schedule D page 2" of the NIST database.

5) Template analysis

By aligning all of the high scoring images in a dataset, a statistical model can be created of the form in question. The areas with high variance can be considered as good candidate regions for crucial information on a form. Figure 7 shows the mean of all of the forms after they have been rotated and translated to line up with each other. Figure 8 shows the areas of high variance for "Schedule D page 2".

Schedule D (Form 1040) 1988 Attachment Sequence No. 12 Page 2

Name(s) as shown on Form 1040. (Do not enter name and social security number if shown on other side.) Your social security number

Part III Summary of Parts I and II

18 Combine lines 8 and 17, and enter the net gain or (loss) here. If result is a gain, also enter the gain on Form 1040, line 13. 18

19 If line 18 is a (loss), enter here and as a (loss) on Form 1040, line 13, the smaller of:
 a The (loss) on line 18; or
 b (\$3,000) or, if married filing a separate return, (\$1,500). 19 ()

Note: When figuring which amount is smaller, treat them as if they were positive numbers.

Part IV Computation of Capital Loss Carryovers From 1988 to 1989
 (Complete this part if the loss on line 18 is more than the loss on line 19.)

Figure 7: Average value of "Schedule D Page 2"

Schedule D (Form 1040) 1988 Attachment Sequence No. 12 Page 2

Name(s) as shown on Form 1040. (Do not enter name and social security number if shown on other side.) Your social security number

Part III Summary of Parts I and II

18 Combine lines 8 and 17, and enter the net gain or (loss) here. If result is a gain, also enter the gain on Form 1040, line 13. 18

19 If line 18 is a (loss), enter here and as a (loss) on Form 1040, line 13, the smaller of:
 a The (loss) on line 18; or
 b (\$3,000) or, if married filing a separate return, (\$1,500). 19 ()

Note: When figuring which amount is smaller, treat them as if they were positive numbers.

Part IV Computation of Capital Loss Carryovers From 1988 to 1989
 (Complete this part if the loss on line 18 is more than the loss on line 19.)

Figure 8: Areas of high variance (dark areas) in "Schedule D Page 2"

This process can be used as a way to bootstrap traditional form image analysis. The only action required by an operator is to give the system a small number of sample images, and then to have a large set of unlabeled documents for more robust statistical analysis. This procedure will then find a robust pair of connected components that will line up the images so a statistical model can be created to capture information that is entered into the form.

6) Future work

This work is a foundation for a more complete form analysis system. It would be an improvement to use not just one pair of connected components but a linear sum of pairs to better align the images. The automatic alignment algorithm is fairly expensive and there are probably several optimizations that could make the alignment much faster. One would be to align a sub-set of the images and then prune out very unlikely matches.

Some forms are much more difficult to interpret than the IRS forms that are in the NIST database. They may have free-flowing fields, or may be severely corrupted by noise. The author believes that this work can be applied to these types of images, but instead of creating just one template, a multi-modal template would have to be created to compensate for the various types of noise and form sub structure.

7) References

[1] *Mitek Products and Services: Doctus* (2003), retrieved 3 March 2003, from Mitek corporate website: http://www.miteksys.com/02_prod_services/sub_sections/document_processing/doctus/doctus.htm

[2] *Product Overview: FormWare* (2003), retrieved 3 March 2003, from Captiva Software corporate website: <http://www.captivasoftware.com/products/formware.asp>

[3] *ReadSoft - Eyes & Hands FORMS* (2003), retrieved 3 March 2003, from ReadSoft's corporate website: <http://www.readsoft.com/products/forms/>

Amplifying Accuracy through Style Consistency

Prateek Sarkar, Thomas Breuel
Palo Alto Research Center
Palo Alto, CA 94034 USA
EMAIL {psarkar,tbreuel}@parc.com
URL <http://www.parc.com/istl/groups/did>

Abstract

Character recognition errors have been reduced by 20-50% in laboratory experiments by style conscious, style specific, and adaptive classification methods. The boost in performance is achieved by exploiting knowledge that the patterns share a common style. Style specific methods are applicable when huge volumes of similar documents are to be recognized. A few documents can be used for training or tuning classifiers which can then be applied to the remaining documents. Document image decoding models can be trained and applied over a wide range of image degradations to achieve error rates of 1% or lower. Hierarchical mixture models capture strong style dependence of patterns across classes. They can be trained automatically from data without style labels. They are specially useful for classifying patterns that are too few to allow parameter tuning or adaptation. Optimal classification of long fields is now practicable due to a new algorithm. A generalization to continuous styles is achieved by a hierarchical Bayesian approach that learns hyperpriors representing a distribution of styles. Adaptive methods cluster large samples of isogenous patterns into equivalence groups, and then assign class labels to these groups. Novel similarity measures improve classification performance.

Several methods of exploiting style consistency for better recognition have been proposed and tested at PARC. Government grants will enable the continuation of this research as well as the building of tools that can be used outside a research environment.

1 Introduction

The multitude of styles in patterns pose significant problems in the design and performance of OCR/DIA systems. Style-specific classifiers that are trained on a single style and applied to the same style perform much better than multi-style classifiers [BN94].

Styles are induced by the source or origin of pat-

terns. **Isogenous patterns** (patterns with a common origin) exhibit traits of the common source resulting in consistency of style, which can be exploited for better recognition. Style consistency is, in fact, very common. Figure 1 shows examples of various styles, and consistency of style in text images. Patterns of the same class show less variability within a style than across styles. Also isogenous patterns are statistically interdependent due to the shared origin (such as the same writer, or font, or degradation, or any combination of these factors).

Style consistency is, of course, not limited to character patterns.

- **Speech** segments in a directory assistance call may come from the same speaker through the same recording device and transmission chan-

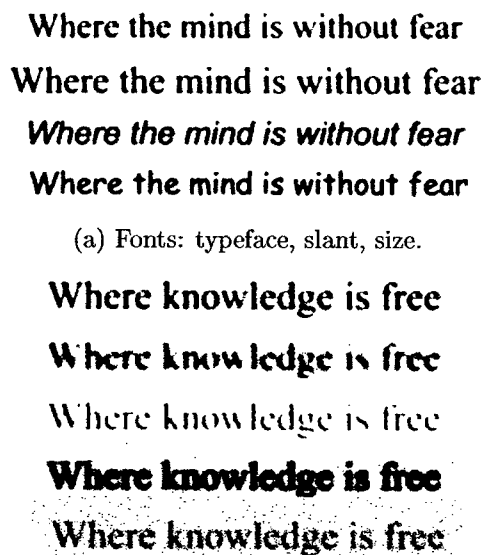


Figure 1: Styles in printed characters.

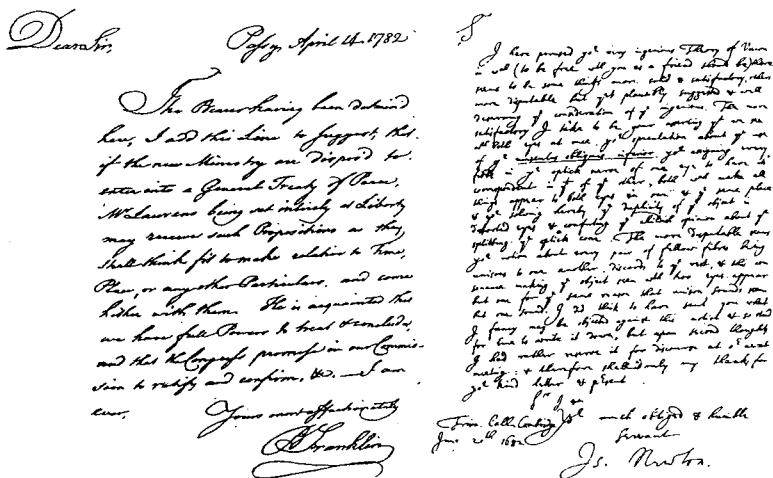


Figure 2: Styles in handwriting.

nels,

- objects in aerial photographs all share similar levels of illumination and viewing angle,
- Two articles/reports encoding the same information reflect the authors' "style" in the choice of words and phrases.

In statistical terms, each style represents a set of class conditional distributions from which patterns are drawn. If the style of a pattern (or field) is known, the best performance is achieved by applying a classifier trained specifically for that style. However, in most applications data to be classified do not come with style labels. If patterns are organized into isogenous fields (as is the case with most OCR applications) performance can still be improved over "style-unaware" classifiers by classifying entire fields in tandem, exploiting the knowledge that all the patterns share a style. Classifiers that model style consistency can be static or adaptive.

Static classifiers are pre-trained to fixed set of styles. Such training can be supervised, as in Optical Font Recognition (OFR), or unsupervised, as in style consistency models. Because of prior training to styles, these systems implicitly or explicitly model **strong style consistency** – the dependence among patterns of different classes (e.g., if the "a" is right slanted, so is the "b"). Including such strong dependence in the model enables these systems to improve performance on short fields, but makes them less suitable for previously unseen styles.

Adaptive classifiers typically tune to the style of the test patterns. The models assume **weak style consistency** constraints, i.e., patterns of the same class are consistent (e.g., all instances of "a" in a document look alike). By making no assumptions about the dependence across classes, these systems

are often able to tune in to previously unseen styles. However, tuning in to a particular style often requires enough samples of each class to be present in the test data.

We summarize recent advances in PARC research on styles, and a few methods of style conscious classification developed by the authors. In laboratory experiments these methods have improved OCR accuracy substantially over other methods of comparable complexity. These methods, including further details and discussions, have been published or submitted for publication elsewhere (references included).

2 Robust style-specific Document Image Decoding¹

Document image decoding (DID) [KC93, KC94, KL96, KL97, Kop97b, Kop97a] is an elegant approach to document recognition from a signal processing perspective. It unifies generative models for a textual message, character shape, text layout, and image degradation into a single generative model for a document image. Given the observed image and knowledge of these generative models, it frames the recognition problem as the maximization of a single probability score for optimal recovery of the original text message. The decoding process therefore obviates the need for separate (often heuristic) steps for image enhancement/de-noising, character segmentation, character recognition, and textual post-processing.

By using character templates for decoding, DID emphasizes the use of document specific models for character shape, layout, and degradation. Given an appropriate choice of templates, DID has been

¹This is a summary of a paper submitted to the International Conference on Document Analysis and Recognition (ICDAR), 2003.

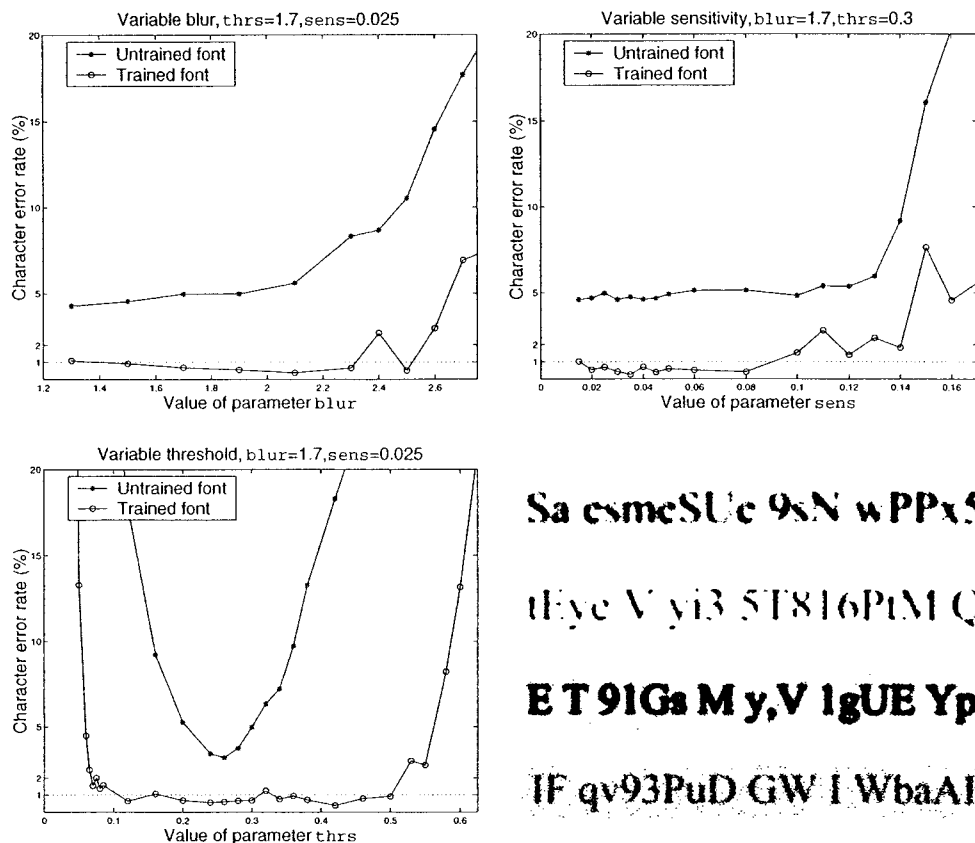


Figure 3: Recognition performance of trained DID models at different points of the degradation parameter space. Also shown are examples of extreme degradation where DID performance still holds at 99% or higher.

shown to deliver extremely high accuracy character recognition results. In practice these models have to be trained on similar documents. Recent work has enhanced the robustness of training algorithms for DID.

Trials were run on synthetic images of text degraded according to the Baird model [Bai92]. The experiments spanned wide ranges of three parameters: *blur* (the standard error of a spherical Gaussian point spread function), *threshold* (against which blurred intensity is compared for binarization into black and white), *sens* (the standard error of a pixel-wise Gaussian additive noise representing sensor error).

Figure 3 shows examples of degradation at the fringes of the parameter ranges that returned high recognition accuracy (~99%) with trained DID models. This highlights the underlying value of DID recognition — very high OCR accuracies can be obtained even when the test images are severely degraded. The only requirement is models trained on similar images, making DID an attractive approach to high volume OCR.

Note that training DID models requires only images of text lines, and their corresponding “ground

truth” transcriptions. The images need not be segmented into character boxes — often an expensive and error prone process that needs considerable manual operation. Our experiments suggest that a few samples of each character is sufficient for training, since the DID degradation model (channel parameters) is learned from the ensemble of *all* character samples.

We summarize advantages of style-specific DID:

1. High accuracy OCR over wide ranges of degradation
2. Inexpensive training data, since character segmentation is unnecessary
3. Academically acclaimed decoding algorithms [MBP01, PBG00, BMP01]
4. Well suited to high volume OCR

3 Fast iterative algorithm for style constrained field classification²

Sarkar *et al.* propose a model of style consistency where pattern features in a field are independently

²Summary of work presented in [Sar02].

distributed when conditioned on class and style, but dependent when conditioning on style is removed. Let x_1, x_2, \dots, x_L be the features measured on L isogenous patterns in a field, and let c_1, c_2, \dots, c_L be the respective class labels. If there are K styles, $k = 1 \dots K$, the combined field feature distribution is modeled as:

$$p(x_1 \dots x_L | c_1 \dots c_L) = \sum_{k=1}^K p_k \prod_{l=1}^L p(x_l | c_l, k) \quad (1)$$

where p_k is the probability that a field derives from style k .

Optimal classification of a field under this model requires the maximization of the left hand side of (1) over all possible sequences of class labels c_1, \dots, c_L . For C classes and a field of length L , this amounts to a maximization over C^L field classes. This exponential complexity makes optimal classification impractical for long fields.

An iterative algorithm proposed in [Sar02] overcomes this difficulty by pruning away unlikely solutions on the basis of a quickly computed upper bound. Laboratory experiments on classifying 4 digit numeral fields yielded a **40% reduction in error rates**, compared to a traditional classifier, while only marginally increasing computational cost. It was observed that, on average, full computation of the likelihood score was being performed for only **25 of 10,000 candidate field labels**, while the others were quickly being eliminated from competition.

This makes the hierarchical mixture model for style consistency practically applicable to longer fields. This is a model of **strong style consistency**, and therefore, can be applied to fields that are too short to allow adaptive methods.

The parameters of the style model can be learned automatically from data without style labels [Sar00], and is therefore also applicable to handprint recognition, where styles are difficult to enumerate.

Summary:

1. Improves recognition accuracy
2. Algorithm speed up does not sacrifice guarantees of optimality
3. Classify short as well as long fields
4. Models automatically trainable [Sar00]

4 A Bayesian framework for style adaptation³

In the hierarchical mixture model for style consistency with a finite number of styles [SN01], each class-style combination (called a class variant)

³Summary of [MB02]

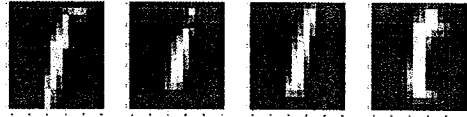


Figure 4: Samples of the letters “f”, “i”, “l”, and “t” after preprocessing.

is represented by a variant-conditional distribution $p(x|\text{class}, \text{style})$, or equivalently, by a class-style specific parameter (vector) $\theta_{\text{class}, \text{style}}$. Mathis and Breuel [MB02] present a generalization with continuous style.

They discuss an integrated Bayesian approach, and as a special case present the hierarchical Bayesian formulation where the class-style conditional distribution parameter θ_ω is drawn from a hyperprior distribution $D^H(\theta_\omega; \Theta_\omega)$ (ω represents the pattern class, Θ_ω is the class-specific hyperparameter).

To enforce style consistency, each class-specific hyperprior is sampled once per field, and all instances of the class are sampled from the resulting class-style conditional distribution. In their experiments the parameter for each class is picked independently of the other (weak style consistency).

Hierarchical Bayesian classification was trained and tested on synthetic pixmaps of digits 0–9 rendered in 12 different fonts, and degraded with different parameter settings of the Baird model [Bai92]. Several batches of samples were generated, each batch with a fixed combination of font and degradation parameters (simulating isogenous fields). Figure 4 shows examples of digits after size normalization to 16×16 . The top seven principal component projections of the resulting 256 pixel-intensities were used in the experimental feature vector.

The results in Table 1 compare the performance of classification by Mixture Discriminant Analysis (which is *style oblivious*) with the *style conscious* hierarchical mixture model of Sarkar *et al.* [SN01] and the hierarchical Bayes method. When training and test sets are identical, both style conscious models perform equally well, and better than the style oblivious model. When the test set includes **fonts unseen during training**, the hierarchical Bayesian method results in best classification.

Summary:

1. Bayesian formulation with continuous styles
2. Matches performance of hierarchical finite mixtures models when test and training fonts are identical.
3. High accuracy even on previously unseen fonts
4. Particularly suited for long fields of isogenous patterns

| training fonts | test fonts | total # training samples | total # test samples | Mixture Discriminant Error | Style Conscious Error | Hierarchical Bayes Error |
|----------------|---------------|--------------------------|----------------------|----------------------------|-----------------------|--------------------------|
| 3,7,10 | 3,7,10 | 600 | 600 | .33% | 0% | 0% |
| 3,7,10 | 3,7,10 | 1500 | 1500 | .20% | .13% | .13% |
| 2,3,6,7,10,11 | 2,3,6,7,10,11 | 1200 | 1200 | .33% | .25% | 0% |
| 1-12 | 1-12 | 2400 | 2400 | .17% | .125% | .08% |
| 3,7,10 | 1,5,9 | 600 | 600 | 1.00% | 2.00% | .08% |
| 3,7,10 | 1,5,9 | 1500 | 1500 | 1.87% | 1.6% | 1.2% |
| 2,3,6,7,10,11 | 1,4,5,8,9,12 | 1200 | 1200 | 1.17% | .50% | .17% |

Table 1: Experimental comparison of style oblivious, style conscious and hierarchical Bayesian classification. Each row of the table is created by generating separate training and test sets with the fonts and sample sizes specified, and applying each of the three classification methods to these two sets. Fonts and sample size cases to report were selected prior to computing experimental error rates. Fonts are Lucida Bright (1-4), Sans (5-8), Typewriter (9-12), in Regular/DemiBold/Italic/DemiItalic variants each.

5 Learning “likeness”: Classification by probabilistic clustering⁴

Under weak style consistency, same-class patterns in a field cluster tightly or show low variability. If patterns in a field can be clustered into equivalence groups, such that all patterns assigned to a group belong to the same class, then the problem of classification reduces to one of assigning class labels to each group. Note that multiple groups could have the same class-label, thus accomodating multiple fonts in the document. This problem can be solved by linguistic clues (cryptogram solving [NSE87]), or by manually labeling representative samples from each cluster. If we also assume that classes do not migrate much under style variations, then it may be

⁴Summary of [Bre01].

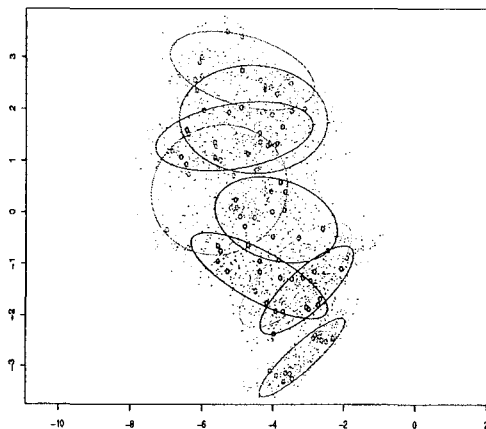


Figure 5: Two sigma plots in the space of the first two principal components of the Gaussian approximations to the distribution of per-batch means for the digits 0-9, derived from 12 fonts.

sufficient to initialize cluster center labels to prior known samples of each class, or bootstrap the system with a modestly competent classifier.

An advantage of this approach is that the classification process is independent of font or style, and works even for previously unseen styles. One such algorithm is presented in [Bre01]. Instead of computing Euclidean distance between feature vectors, the clustering performance is improved by a novel similarity function: $P(c = c', f = f' | v, v')$, the probability that two patterns (feature vectors) v, v' derive from the same class and font. Note that this model does not depend on the true class or font identities – only on whether they are the same or not. This probabilistic model is trained from pairs of labeled patterns. In experiments a multi-layer perceptron is trained to estimate these probabilities from a pair of input patterns. Test patterns are then clustered using this similarity function. Experimental data (bitmaps of the 10 digit classes) are artificially gen-

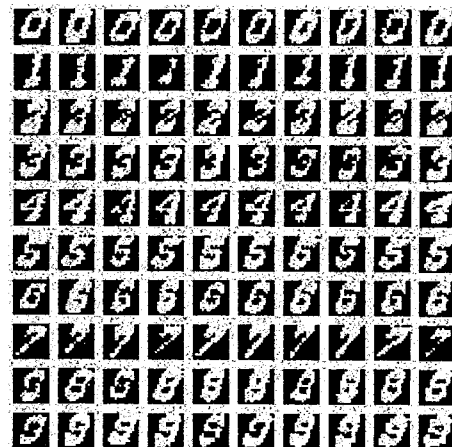


Figure 6: Examples of degraded characters.

erated by digitizing TrueType fonts through Baird's degradation model (Figure 6). 717 fonts, 10 samples per class, per font were used in the experiment. 646 of these fonts were used to train the multi-layer perceptron, and 71 other fonts were used for testing. The error rate for this method was 7.66% – 20% lower (relative) than that of a traditional MLP classifier trained to output character classes directly.

Summary:

1. No prior style models necessary
2. Novel similarity measure improves clustering performance
3. Easily extendable to new fonts, and new characters, new languages

6 Conclusion

Style consistency and adaptation models presented above can be applied to many pattern recognition and data mining problems. Character recognition is just one of them. Due to the relative ease of data collection/generation, and the various intuitive sources of style OCR has been an excellent test bed for these ideas.

Government sponsorship will enhance our ability to do further research in these directions as well as wrap developed ideas into computer code useable outside the research domain.

References

- [Bai92] H. S. Baird. Document image defect models. In H. S. Baird, H. Bunke, and K. Yamamoto, editors, *Structured Document Analysis*, pages 546–556. Springer-Verlag, New York, 1992.
- [BMP01] Dan S. Bloomberg, Thomas P. Minka, and Kris Popat. Document image decoding using iterated complete path search with subsampled heuristic scoring. In *Proceedings of the IAPR 2001 International Conference Document Analysis and Recognition (ICDAR 2001)*, September 2001.
- [BN94] H. S. Baird and G. Nagy. A self-correcting 100-font classifier. In L. Vincent and T. Pavlidis, editors, *Document Recognition, Proceedings of the SPIE*, volume 2181, pages 106–115. 1994.
- [Bre01] T.M. Breuel. Classification by probabilistic clustering. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (ICASSP 2001)*, pages 1333–1336, 2001.
- [KC93] Gary Kopec and Phil Chou. Automatic generation of custom document image decoders. In *Proceedings of the Second Intl. Conf. on Document Analysis and Recognition*, Tsukuba Science City, Japan, Oct 1993.
- [KC94] G. Kopec and P. Chou. Document image decoding using markov source models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 16(6):602–617, June 1994.
- [KL96] G. Kopec and M. Lomelin. Document-specific glyph template estimation. In L. Vincent and J. Hull, editors, *Proceedings of Document Recognition III, SPIE, vol. 2660*, pages 14–26, 1996.
- [KL97] G. E. Kopec and M. Lomelin. Supervised template estimation for document image decoding. *IEEE Trans. on PAMI*, 19(12):1313–1324, December 1997.
- [Kop97a] G. Kopec. Em estimation of templates. Unpublished manuscript, Xerox PARC, 1997.
- [Kop97b] G. Kopec. Multilevel character templates for document image decoding. In L. Vincent and J. Hull, editors, *Document Recognition IV, Proceedings of SPIE, vol. 3027*, 1997.
- [MB02] C. Mathis and T. Breuel. Classification using a hierarchical Bayesian approach. In *Proceedings of the Sixteenth ICPR*, Quebec City, Canada, August 2002.
- [MBP01] Thomas P. Minka, Dan S. Bloomberg, and Kris Popat. Document image decoding using the iterated complete path heuristic. In *Proceedings of IST/SPIE Electronic Imaging 2001: Document Recognition and Retrieval VIII*, January 2001.
- [NSE87] G. Nagy, S. Seth, and K. Einspahr. Decoding substitution ciphers by means of word matching with application to OCR. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (9):710–715, 1987.
- [PBG00] Kris Popat, Dan Bloomberg, and Dan Greene. Adding linguistic constraints to document image decoding. In *Proceedings of the 4th IAPR Workshop on Document Analysis Systems (DAS 2000)*, December 2000.
- [Sar00] Prateek Sarkar. *Style consistency in pattern fields*. PhD thesis, Rensselaer Polytechnic Institute, Troy, NY, USA, 2000.
- [Sar02] Prateek Sarkar. An iterative algorithm for optimal style-conscious field classification. In *Proceedings of the sixteenth ICPR*, volume IV, pages 243–246, Quebec City, Canada, August 2002. IEEE Computer Society Press.
- [SN01] Prateek Sarkar and George Nagy. Style consistency in isogenous patterns. In *Proceedings of the Sixth ICDAR*, pages 1169–1174, Seattle, USA, September 2001.
- [VN02] Sriharsha Veeramachaneni and George Nagy. Classifier adaptation with non-representative training data. In *Proceedings of Document Analysis Systems*, pages 123–133, 2002.

Multimedia

Form Analysis with the Nondeterministic Agent System (NDAS)

Thomas C. Henderson and Lavanya Swaminathan
School of Computing
University of Utah
Salt lake City, Utah 84112, USA
tch@cs.utah.edu

February 27, 2003

Abstract

We have been developing a framework within which multiple agents cooperate to analyze scanned images of engineering drawings. While our main focus has been on extracting dimension annotations from CAD drawings, we describe here how our method can be applied to form processing. The basic ideas behind our approach are (1) allow the nondeterministic exploration of a large search space, and (2) exploit structural constraints to rapidly prune the search space.

1 Introduction

The complete analysis of engineering drawings is still an unsolved problem. Most extant systems exhibit brittle performance when applied to real world image sets. (See Tombre [1] for an overview of the field and Kanungo et al. [2] for insightful commentary.) An approach very similar to ours is that of Bapst et al. [3]. They propose a system for Cooperative and Interactive Document Reverse Engineering (CIDRE). Its major features are that it:

- is human-assisted,
- provides automatic inference of document models,
- has a cooperative, multi-agent architecture,
- explores the search space concurrently,
- commits to an interpretation only when confidence is high, and
- provides for local revisions of belief.

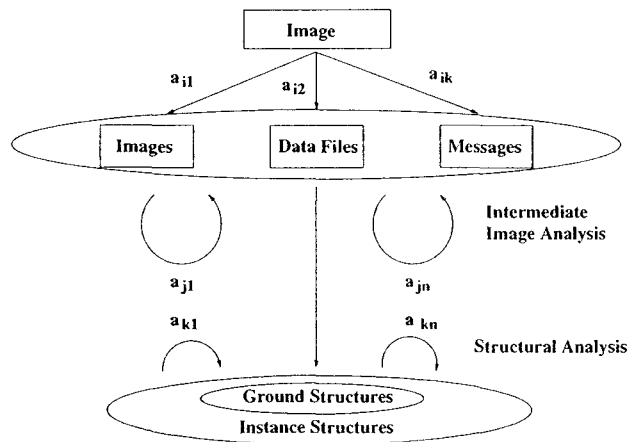


Figure 1: NDAS Agent Organization

We have developed a similar approach called the NonDeterministic Agent System (NDAS). NDAS shares some of CIDRE's philosophy; however, as the authors of CIDRE note: "one of the main drawbacks in our proposition is the present lack of a high-level reasoning language that can express intelligent behavior." Our view is that this can be addressed by putting into place a mechanism to control the combinatorics of the search space growth. We achieve this through the exploitation of structural constraints in the underlying document, and through feedback mechanisms to reinforce paths that lead to successful interpretations. See Swaminathan [4] for details on the NDAS approach as applied to technical drawing analysis.

2 Nondeterministic Agents

NDAS is basically a blackboard system with a set of independent agents, each with expertise on some particular aspect of the analysis problem. In addition, there are agents that monitor the form analysis as it proceeds and give feedback to reinforce successful results. (See Figure 1.)

An agent is a software process that:

- is autonomous: runs independently of other agents;
- has state: assertions, relations memory;
- is persistent: never terminates;
- can communicate: send and receive messages related to the analysis;
- can sense: is aware of or can detect relevant input or conditions;
- and can act: analyze or create information.

For more complete accounts, see [5,6].

An agent architecture is a software architecture for decision making with intelligent (flexible) processes embedded within it. The agents may be proactive or reactive, and should cooperate (including communicate) to achieve a goal.

We explore the use of nondeterministic agent systems (NDAS) to achieve a more flexible system for technical drawing analysis. They are called nondeterministic because the agents explore alternative parts of the solution space simultaneously, and every agent works to produce some result which may or may not contribute to the final result. The final result derives from only a subset of the work put in by all the agents. We explore nondeterminism in this problem domain since deterministic systems usually make irrevocable decisions (e.g., threshold selection) that eliminate possible solutions. The technical drawing problem domain contains many factors that vary with the drawing: thresholds, text fonts and size, noise levels, etc., and this variation makes it interesting to explore the possible solution space dynamically and in a breadth-first way.

We demonstrate this through the design and analysis of the NDAS system and provide experimental results to support the claims. (See [4] for more detail.)

3 Form Structure and Constraints

Forms may be described in terms of the layout and relations between the logical parts of the form. Several methods have been previously proposed to perform document structure analysis and to identify form classes; e.g., [7] describes a document analysis system to analyze forms from the French social services department (les Allocations Familiales). Others have reported on a variety of issues: Wang and Shirai [8] discuss a bottom-up approach that segments boxes, line segments and text into components, determines spatial relationships between boxes and text, and separates text from lines that touch or overlap. They give several examples of removing lines. Ha and Bunke [9] demonstrate good results of model-based analysis and understanding of checks; Arai and Odaka [10] perform background region analysis and extract box regions; they tested 50 types of forms; Tang and Lin [11] show how semantic information can be acquired and forms efficiently stored.

For our work here, we assume that a form model has been acquired and is expressed as a structural model including spatial relation constraints. For example, the box structure of the form in Figure 2 could be described as:

```
Rule 1: form := A + B + C
        [side(A,1)=side(B,3);
         len(A,2)=len(B,2)]
```

```
Rule 2: A := box
```

```
Rule 3: B := box1 + box2
```

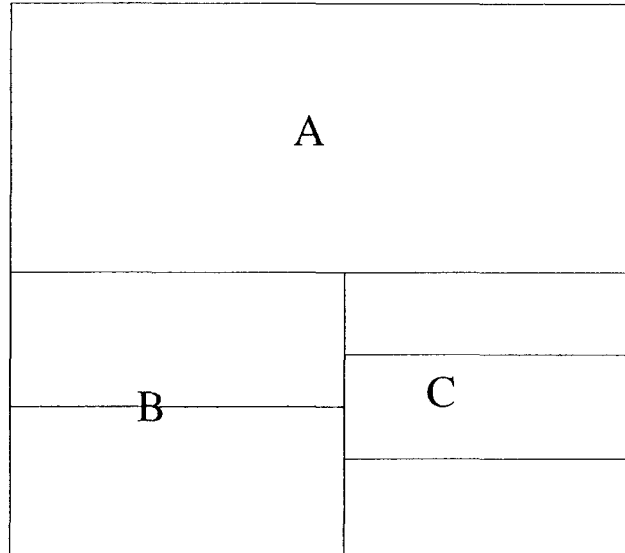


Figure 2: Boxes Example

```
[side(1,3)=side(2,1);
len(1,2)=len(2,2)]
```

```
Rule 4: C := box1 + box2 + box3
[side(1,3)=side(2,1);
side(2,3)=side(3,1);
len(1,2)=len(2,2);
len(1,2)=len(3,2)]
```

Simple rectangles form the lowest level terminal symbols; *A* is a nonterminal representing the top box; *B* represents the lower left 2 boxes; *C* represents the lower right 3 boxes; *form* represents *A*, *B*, and *C* in their proper configuration. The sides of boxes are numbered lower, right, upper, and left as 1, 2, 3 and 4, respectively. The square brackets in the rules contain the constraints as predicates about the geometry of the boxes.

We have done a number of detailed experiments on the analysis of technical drawings [4]. As for the analysis of document forms, we have only performed an initial study at the request of the SDIUT organizers, and thus, can only report on our first cut at using NDAS on this problem. The initial results are encouraging.

NDAS already has image analysis agents for the first steps in form processing: thresholding, edge and box extraction; we have added agents to perform the structural analysis described in the rewrite rules given above. If there are only terminal symbols (simple boxes) on the right hand side of the rewrite rule, then the agent looks for simple boxes produced directly from the image analysis; if they satisfy the required relations (if any), then an instance of the left hand

5. Evaluation Profile: please check the appropriate boxes.

| | Outstanding | Unusual | Good | Average | Below Average | Unknown |
|---------------------------------------|-------------|---------|------|---------|---------------|---------|
| Communication skills (oral & written) | | | | | | |
| Motivation | | | | | | |
| Analytical thinking | | | | | | |
| Independence | | | | | | |
| Creativity | | | | | | |
| Ability to work with others | | | | | | |

Figure 3: A Graduate Reference Form

side of the rule is produced. Otherwise, the agent watches for instances of the nonterminals to be created and then goes to work once they appear.

Figure 3 shows part of the form that we use in the School of Computing in our graduate admissions application process; we would like to automatically extract the information from these legacy forms. We hope to report at the meeting more detailed results of the application of NDAS to the understanding of such forms.

4 Conclusions and Future Work

We have demonstrated elsewhere that the NDAS approach combined with structural pruning methods can be very advantageous [4] for the analysis of technical drawings. Here we have presented some preliminary data on the application of the NDAS approach to document form analysis. A coherent analysis is performed which allows for a wide range of form structures as well as the possibility to explore the large search space necessary for a robust result.

The next steps include:

- develop a more comprehensive document form model,
- perform more in-depth experiments on a wider class of documents,
- investigate feedback mechanisms from higher-level analysis agents to the image analysis and structural parsing agents.

Acknowledgements

This work was supported in part by ARO grant number DAAD19-01-1-0013. Thanks to Jim de St. Germain for help with form document scanning.

References

- [1] K. Tombre. Analysis of engineering drawings: State of art and challenges. In *Graphics Recognition - Algorithms and Systems*, volume 1389 of *Lecture Notes in Computer Science*, pages 257–264, Springer Verlag, April 1998.

- [2] T. Kanungo, R. M. Haralick, and D. Dori. Understanding engineering drawings: A survey. In *Proceedings of First IARP Workshop on Graphics Recognition*, pages 217–228, University Park, P.A, 1995.
- [3] F. Bapst, R. Brugger, A. Zramdini, and R. Ingold. Integrated multi-agent architecture for assisted document recognition. In *Document Image Analysis*, pages 301–317, World Science, Singapore, 1994.
- [4] L. Swaminathan. Agent-based engineering drawing analysis. Master’s thesis, University of Utah, Salt Lake City, Utah, December 2002.
- [5] V. Subrahmanian. *Heterogeneous Agent Systems*. MIT Press, Cambridge, MA, 2000.
- [6] G. Weiss. *Multi-Agent Systems*. MIT Press, Cambridge, MA, 1999.
- [7] S. Diana, E. Trupin, Y. Lecourtier, and J. Labiche. Document modeling for form class identification. In *Advances in Document Image Analysis*, pages 176–187, Springer-Verlag, Berlin, 1997.
- [8] D. Wang and S. Shirai. Analysis of form images. In *Document Image Analysis*, pages 35–56, World Science, Singapore, 1994.
- [9] T. Ha and H. Bunke. Model-based analysis and understanding of check forms. In *Document Image Analysis*, pages 57–84, World Science, Singapore, 1994.
- [10] H. Arai and K. Okada. Form processing based on background region analysis. In *Proc ICDAR 97*, pages 164–169, Ulm, Germany, 1997.
- [11] Y. Tang and J. Lin. Information acquisition and storage of forms in document processing. In *Proc ICDAR 97*, pages 170–174, Ulm, Germany, 1997.

Metrics for Evaluating the Performance of Video Text Recognition Systems

Gregory K. Myers
SRI International
333 Ravenswood Avenue
Menlo Park, California 94025
gregory.myers@sri.com

Abstract

Video is an increasingly important source of information to the intelligence analyst. The recognition of text that appears in video imagery is potentially useful for automatically characterizing its content. Evaluating the performance of video text recognition systems is crucial for predicting the behavior of a system applied to specific classes of video data, comparing one system with another, and measuring development progress over time. In video imagery the same text can appear in multiple consecutive frames; hence, the metrics required for OCR in video are somewhat different from those required for OCR in documents and still imagery. We develop a set of metrics that are defined relative to a text object, a unit of text that has an extent in both time and location in the imagery. We discuss the relative importance of the individual metrics and their trade-offs for particular applications of video text recognition. These metrics take into account partial recognition and imperfect segmentation for multiple recognition results across multiple frames. The metrics are used to evaluate the performance of a video text recognition process developed at SRI International.

1 Introduction

Video is an increasingly important source of information to the intelligence analyst. Recognizing text that appears in video imagery is potentially useful for automatically characterizing its content. Evaluating the performance of video OCR or video text recognition (VTR) systems is crucial for predicting the behavior of a system applied to specific classes of video data, because the performance of VTR systems is highly dependent on video image quality, imaging configuration (viewing angles, camera motion, etc.), and types of content. It is also useful for comparing one system with another, and measuring development progress over time. VTR processes have been applied to such applications as the indexing and retrieval of information from video archives [1,2,3,4,5] and license plate reading [6,7,8]. In each of these applications, the VTR capability is a functional module that is part of a larger automated or semi-automated system designed to

perform a specific set of tasks. Therefore, metrics should be developed that are relevant to the tasks and the modes in which the VTR process is used. To this end we have developed a set of metrics that can be used to characterize the performance of all video text recognition systems; the relative importance of each of the metrics depends on the specific task or application for which the VTR system is used.

2 Major Issues

Two major issues are associated with the definition of metrics that assess the overall performance of VTR systems. The first issue pertains to the fact that “text” can refer to an individual character, a word, a line of text, or even a block of text lines (either geometrically or logically related). Ideally, the unit of text used for evaluating recognition performance should be the one that is most meaningful for the particular application. The use of a character as the unit of text is the most common way that OCR systems have been evaluated. However, the effort to truth the position of every character in each frame is extremely burdensome. On the other hand, if an instance of text is defined as anything larger than a character, methods of evaluation will have to contend with questions related to partial recognition (the correct recognition of some but not all components) and imperfect segmentation (splitting or merging of components). For example,

- How should a result like “MENLOPARK” (versus “MENLO PARK”) on a sign be scored?
- How should punctuation be treated?
- How should false characters at ends of an otherwise completely correct word (e.g., “!1Parking”) be treated?
- How should the segmentation of entries into rows or columns on a posted schedule be treated?

The second issue pertains to the nature of time in video. Because in video imagery the same text can appear in multiple consecutive frames, each instance of text in a video has a start and end time associated with its appearance. The ideal recognition process would produce one correct recognition result for each instance of text. Partial recognition and segmentation across

time can also be a problem. For example,

- If the text “MENLO PARK” is present in the video from frame 100 to frame 200, how should we score a result that says the text was present from frame 101 to frame 160?
- How should we score the results when “MENLO PARK” is recognized from frame 100 to 160, and “MENLO PHAK” is recognized from frame 161 to 200?
- How should we score the results when the recognition process produces not one correct result for the entire frame interval, but a multiplicity of correct results, each associated with much shorter time intervals (one correct result for frames 100 to 109, another correct result from frames 110 to 120, etc.)?

The definition of a single instance of text may itself be open to interpretation. For example, what should be included in the ground truth data when a camera views some text for a period, turns away so that the text is out of its field of view, and then turns back again? Should we say that these are two separate instances of text? What if a moving vehicle obscures the text for a few frames? What if only some of the text is obscured at any one time (such as would happen if someone were to walk in front of a large sign on the side of a truck)?

Metrics for end-to-end VTR systems have included the fractions of correctly and erroneously recognized characters, and the frequency of false characters [1,2,5]. These quantities have also been expressed in terms of recall and precision rates, which are convenient measures when the VTR system is used for indexing and retrieving video segments [3,4,9,10]. Metrics have also been developed for evaluating various component processes of VTR (such as text detection and text tracking); these include measures of detection and false alarm (or recall and precision), spatial and temporal localization, and spatial and temporal fragmentation [11,12,13]. However, none of these previous works has directly addressed the issue of handling multiple recognition results across multiple frames.

3 Metrics

To define our desired metrics, we first define some terms related to the comparison of recognized text with ground truth data.

Central to our evaluation approach is the concept of a *text object*, a unit of text that has an extent both in time and in its location in the imagery, defined respectively by the following two terms. The *frame range* of text object X_i is an interval $[f_{si}, f_{ei}]$, where f_{si} and f_{ei} are the starting and ending frame numbers, respectively; the *image volume* of X_i is $v_i = \{a_f\}$, where a_f is the area occupied by the text object in frame f , and $f_{si} = f = f_{ei}$. To simplify the representation of a text object, we assume that its text content is constant throughout its

duration, even if the text is partially obscured during a portion of its duration. (Text that appears, disappears completely, and then reappears is counted as two separate text objects).

$\{T_i\}$, $i = 1, 2, \dots, N_T$ is the set of *true text objects* appearing in the video.

$\{R_j\}$, $j = 1, 2, \dots, N_R$ is the set of *reported text objects* recognized by the VTR process in the video.

True text objects and reported text objects are defined in terms of the same unit of text (character, word, line, or block). The choice depends on the unit's appropriateness to the task and the form of the available ground truth.

Because of imperfect text tracking or discrepancies in a segmentation, a true text object may produce more than one reported text object. Similarly, a single reported text object may correspond with more than one true text object. If f_i and f_j are the frame ranges of T_i and R_j , respectively, and if $(f_i \cap f_j) / \min(|f_i|, |f_j|) >$ some threshold, T_i and R_j *overlap significantly in frame range*. The frame range of R_j may extend beyond that of T_i because of incorrect linking of separate true reported text objects by the text tracking process, or because R_j is linked with false text. Similarly, if v_i and v_j are the image volumes of T_i and R_j , respectively, and if $(v_i \cap v_j) / \min(|v_i|, |v_j|) >$ a threshold, T_i and R_j *overlap significantly in image volume*. The image volumes of T_i and R_j may differ because of variations in text detection and segmentation, or because of the inclusion of false text within R_j . We say that R_j and T_i *correspond* if their frame ranges and areas significantly overlap.

Let $S(T_i) = \{R_k\}$, $k = 1, 2, \dots, N_i$ be the set of reported text objects that correspond with T_i . A true text object is said to be *missed* if it does not have any corresponding reported text objects, i.e., $S(T_i)$ is the empty set. Conversely, a recognition result is said to be *false* if it does not have any corresponding true text. We say that R_k *matches* T_i if their text contents are identical or have a similarity score beyond a certain threshold. We say that T_i is *correctly recognized* or *correct* if it matches at least one of its corresponding reported text objects; otherwise, it is said to be *wrong*. R_j is said to be *correct* if it matches some corresponding T_i ; otherwise, it is said to be *wrong*.

The *verbosity* V_i of the reported text objects $S(T_i)$ is $|S(T_i)|$, the number of elements in $S(T_i)$. The *coverage* C_i of T_i by its corresponding reported text objects is $|\bigcup_{k=1}^{N_i} f_k| / |f_i|$, the fraction of the frame range $[f_{si}, f_{ei}]$ that is covered by the union of the frame ranges of all the members of $S(T_i)$. The ideal recognition process would produce a single reported text object for T_i (i.e., a verbosity of 1) with a coverage of 100%.

Some quantities to be computed for our metrics are

N_T = the total number of true text objects
 N_R = the total number of reported text objects
 $N_{T\text{-correct}}$ = the number of correct true text objects
 $N_{T\text{-wrong}}$ = the number of wrong true text objects
 $N_{T\text{-miss}}$ = the number of missed true text objects
 $N_{R\text{-correct}}$ = the number of correct reported text objects
 $N_{R\text{-wrong}}$ = the number of wrong reported text objects
 $N_{R\text{-false}}$ = the number of false reported text objects.

The relationships between these quantities are as follows:

$$N_T = N_{T\text{-correct}} + N_{T\text{-wrong}} + N_{T\text{-miss}}$$

$$N_R = N_{R\text{-correct}} + N_{R\text{-wrong}} + N_{R\text{-false}}$$

The metrics are computed as follows:

$M_1 = N_{T\text{-correct}} / N_T$ is the fraction of true text objects that are correct.

$M_2 = N_{T\text{-miss}} / N_T$ is the fraction of true text objects that are missed.

$M_3 = N_{R\text{-correct}} / N_R$ is the fraction of reported text objects that are correct.

$M_4 = N_{R\text{-false}} / N_R$ is the fraction of reported text objects that are false.

$$M_5 = \left(\sum_{i=1}^{N_T - N_{T\text{-miss}}} V_i \right) / (N_T - N_{T\text{-miss}}) = (N_R - N_{R\text{-false}}) / (N_T - N_{T\text{-miss}})$$

$(N_T - N_{T\text{-miss}})$ is the average verbosity.

$$M_6 = \left(\sum_{i=1}^{N_T - N_{T\text{-miss}}} C_i \right) / (N_T - N_{T\text{-miss}})$$

is the average coverage.

The metrics M_1 , M_2 , M_5 , and M_6 are performance measures that are relative to the ground truth. M_3 and M_4 are measures that are relative to the reported text. If the reported text objects were used in an indexing and retrieval task, M_1 would be analogous to the recall rate, and M_3 would be analogous to the precision rate. Because the location and extent of recognized text in the imagery has not been important for VTR applications, we have not developed metrics to characterize the accuracy of these quantities. If necessary, such metrics as those used to evaluate text detection [13] could be applied.

A toy example that illustrates the metrics is shown in Figure 1. The metrics are calculated for three units of text: a line, a word, and a character. We used string edit distance (without transposition) as the similarity measure, with a matching threshold of 0.9.

| Ground Truth Text | | | Reported Text | | |
|-------------------------------------|---------------|---------------|---------------|-------|--------------|
| f_s | f_e | Text Content | f_s | f_e | Text Content |
| 1 | 10 | MENLO PARK | 2 | 5 | MENLO PARK |
| | | | 7 | 7 | MENLO PHAK |
| | | | 9 | 10 | MENLO PARK |
| 1 | 10 | TRAIN STATION | 1 | 9 | XX TRAIN |
| | | | 1 | 4 | STATION |
| 1 | 10 | PARKING LOT | 1 | 10 | YPARKING LOT |
| 1 | 10 | ONE WAY | | | |
| | | | 1 | 2 | ZZZ |
| If the unit of text is a line: | | | | | |
| $N_T = 4$ | $M_1 = 2/4$ | $M_3 = 3/7$ | $M_5 = 6/3$ | | |
| $N_R = 7$ | $M_2 = 1/4$ | $M_4 = 1/7$ | $M_6 = .83$ | | |
| If the unit of text is a word: | | | | | |
| $N_T = 8$ | $M_1 = 5/8$ | $M_3 = 7/12$ | $M_5 = 10/6$ | | |
| $N_R = 12$ | $M_2 = 2/8$ | $M_4 = 2/12$ | $M_6 = .82$ | | |
| If the unit of text is a character: | | | | | |
| $N_T = 37$ | $M_1 = 31/37$ | $M_3 = 47/55$ | $M_5 = 49/31$ | | |
| $N_R = 55$ | $M_2 = 6/37$ | $M_4 = 6/55$ | $M_6 = .75$ | | |

Figure 1: Toy example of video text recognition results.

The reported text has one misspelled word (“PHAK”), one false text line (“ZZZ”), one extra word (“XX”), and one extra character attached to a word (“YPARKING”). Also, the line containing “MENLO PARK” is fragmented into three frame ranges, and the line containing “TRAIN STATION” is segmented into two individual lines. The unit of text used for truing affects how certain results are evaluated. When the unit of text is larger than an individual character, there may be text on either end of a true text object T_i that is missing from $S(T_i)$, or, conversely, some R_k may have some text on either end that has no correspondence in T_i . This text is not counted as missing or false text, but instead has the effect of lowering the similarity score for that $T_i - R_k$ correspondence. For example, the “XX” in “XX TRAIN” is considered to be false text when the unit truing is a word or character, but not when the unit of truing is a line. Similarly, the ‘Y’ in “YPARKING” is considered to be false text only when the unit of truing is a character. The choice of the unit of truing also affects the impact of over-segmenting text regions into multiple components. For example, the phrase “TRAIN STATION” was reported as two separate text lines instead of a single line. If the unit of truing is a line, neither reported result matches the ground truth, and its verbosity is 2 instead of 1. However, if the unit of truing is a word or character, the reported results are considered to be correct.

4 Relative Importance of the Metrics

The relative importance of the six metrics described above depends on the processing mode in which the VTR process is used, and the specific task or

Table 1: Performance of SRI VTR process with $MinFrames = 10$.

| Video | Number of Frames | N_T | N_R | M_1 | M_2 | M_3 | M_4 | M_5 | M_6 |
|-------------------------|------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Café Barrone | 390 | 2 | 8 | 0.50 | 0.00 | 0.12 | 0.50 | 2.00 | 0.71 |
| Caution-Iron | 450 | 16 | 11 | 0.31 | 0.56 | 0.64 | 0.00 | 1.43 | 0.38 |
| Fed Storage | 119 | 3 | 6 | 0.33 | 0.67 | 0.33 | 0.00 | 3.00 | 0.52 |
| 15 TH Avenue | 149 | 2 | 10 | 0.50 | 0.00 | 0.60 | 0.00 | 5.00 | 0.63 |
| Innovation | 161 | 7 | 20 | 1.00 | 0.00 | 0.65 | 0.15 | 2.43 | 0.73 |

Table 2: Performance of SRI VTR process with $MinFrames = 3$.

| Video | Number of Frames | N_T | N_R | M_1 | M_2 | M_3 | M_4 | M_5 | M_6 |
|-------------------------|------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Café Barrone | 390 | 2 | 32 | 0.50 | 0.00 | 0.03 | 0.84 | 2.50 | 0.72 |
| Caution-Iron | 450 | 16 | 39 | 0.62 | 0.25 | 0.64 | 0.03 | 2.92 | 0.33 |
| Fed Storage | 119 | 3 | 8 | 0.67 | 0.00 | 0.37 | 0.00 | 2.67 | 0.28 |
| 15 TH Avenue | 149 | 2 | 17 | 0.50 | 0.00 | 0.41 | 0.24 | 6.50 | 0.70 |
| Innovation | 161 | 7 | 22 | 1.00 | 0.00 | 0.59 | 0.23 | 2.43 | 0.73 |

application. We distinguish two major classes of processing modes: indexing for later retrieval, and immediate triggering of automated action. The former typically involves the generation of content-based indices that are stored along with the video imagery. Later on, if a query matches one or more of the indices, the metadata and/or the associated video segments are retrieved and either processed further or viewed in the form of video clips, key frames, or montages of text regions. The latter processes typically do not involve storage of the imagery, but initiate additional automated processes such as the lookup of a license number in a database, or the tracking and classification of a vehicle bearing a recognized license plate.

The metrics M_1 and M_2 are quite important for both classes of processing modes. M_3 , M_4 , and M_5 may not be as important for a video indexing and retrieval task if the costs of storing and later viewing of the imagery are not significant. M_6 may be less important for indexing and retrieval if the true duration of the reported text objects in the video is short, but may be more important if the process tries to select a single key frame that best represents the entire frame range in which the text object appears. For the immediate triggering of automated action, M_3 , M_4 , and M_5 are important if the cost of the triggered processing is high. M_5 and M_6 may be important in surveillance applications in which the continuous tracking of objects and the times when they enter and leave the field of view are important. On the other hand, M_5 and M_6 may be less important in an application in which the recognized text is used as a query into a database (e.g., a list of license plates of vehicles known to be suspicious) and only the first "hit" in that database is sufficient.

5. Evaluation of the SRI VTR Algorithm

We evaluated a VTR algorithm previously developed at

SRI International for scene text on a set of video segments collected with a hand-held digital camcorder and recorded on mini-DV tape. The video data was saved as 640×480 gray-scale AVI files at 12 frames per second, with no compression other than that produced by the camcorder in the recording process. Tables 1 and 2 list the performance of the VTR process on the collected video segments. The video segments, recognition results, and truth data are available at http://www.erg.sri.com/automation/video_recog.html.

We limited our evaluation to human-readable scene text that is not highly stylized or that is part of a logo (reported results generated from text that does not meet this criteria were ignored). The unit of text for this evaluation was defined as an individual line of text. To be considered correct, the reported text must match at least 90% of the characters in the ground truth.

The performance is highly dependent on the specific characteristics of the text (e.g., contrast, size) and the imaging conditions (e.g., motion blur). The parameter $MinFrames$ controls the minimum number of frames that the VTR process accepts as reported text; text sequences with fewer frames are thought to be false text and are discarded. Table 1 shows performance with $MinFrames = 10$, and Table 2 shows performance with $MinFrames = 3$. Comparing the two tables, we see that reducing $MinFrames$ from 10 to 3 increases the fraction of text lines correctly recognized (M_1) and decreases the fraction of missed text lines (M_2) for video segments that have a high amount of camera motion (e.g., Caution-Iron and Fed Storage). However, this reduction of $MinFrames$ causes the false text rate (M_4) to increase in the other video segments.

6 Summary

Evaluating the performance of video text recognition systems is crucial for predicting the behavior of a system applied to specific classes of video data,

comparing one system with another, and measuring development progress over time. In video imagery, the same text can appear in multiple consecutive frames; hence, the metrics required are somewhat different from those required for OCR in documents and still imagery. We have developed a set of metrics that are defined relative to a *text object*, a unit of text that has an extent in both time and location in the imagery. The metrics take into account partial recognition and imperfect segmentation for multiple recognition results across multiple frames. The relative importance of the individual metrics differs according to the specific task or application for which the VTR system is used.

Acknowledgements

This material is based upon work supported in whole or part by the Advanced Research and Development Activity (ARDA). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Government.

References

- [1] Sato, S., T. Kanade, E. Hughes, and M. Smith, Video OCR for digital news archive, *Proc. 1998 Int'l. Workshop on Content-Based Access of Image and Video Databases (CAIVD '98)*, Bombay, India, IEEE Computer Society, Jan. 1998.
- [2] J. Xi, X.-S. Hua, X.-R. Chen, L. Wenyin, and H.-J. Zhang, A video text detection and recognition system, *2001 IEEE Int. Conf. on Multimedia and Expo (ICME 2001)*, Waseda University, Tokyo, Japan, August 2001, 1080–1083.
- [3] A. Hauptmann, R. Jin, N. Papernick, D. Ng, Y. Qi, R. Houghton, and S. Thornton, Video retrieval with the Informedia Digital Video Library system, *Proc. Tenth Text Retrieval Conf. (TREC-2001)*, Gaithersburg, Maryland, Nov. 2001.
- [4] C.-J. Lin, C.-C. Liu, and H.-H. Chen, A simple method for Chinese video OCR and its application to question answering, *Computational Linguistics and Chinese Language Processing* 6:2 (2001), 11–30.
- [5] R. Lienhart, Localizing and segmenting text in images and videos, *IEEE Trans. Circuits and Systems for Video Technology* 11:4, April 2002.
- [6] P.G. Williams, H.R. Kirby, F.O. Montgomery, and R.D. Boyle (1988), "Evaluation of video-recognition equipment for number-plate matching," *Proc. PTRC Annual Meeting*, P306, University of Leeds, UK 229–239.
- [7] P. Castello, C. Coelho, E. Del Ninno, E. Ottaviani, and M. Zanini, Traffic monitoring in motorways by real-time number plate recognition, *Tenth Int. Conf. Image Analysis and Processing*, Venice, Italy, Sept. 1999.
- [8] Y. Cui and Q. Huang, Character extraction of license plates from video, *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico (1997) 502–507.
- [9] R. Houghton, Named faces: putting names to faces, *IEEE Intell. Systems*, Sept. 1999.
- [10] R. Kasturi, S. Antani, D.J. Crandall, and V.Y. Mariano, A framework for reliable text based indexing of video, *Proc. Symp. on Document Image Understanding Technology*, April 2001.
- [11] H. Li, D. Doermann, and O. Kia, "Automatic Text Detection and Tracking in Digital Video," *IEEE Trans. Image Processing—Special Issue on Image and Video Processing for Digital Libraries* (1999) 147–155.
- [12] X.-S. Hua, L. Wenyin, and H.-J. Zhang, Automatic performance evaluation for video text detection, *Proc. Int. Conf. on Document Analysis and Recognition* (2001).
- [13] V. Mariano, J. Min, J. Park, R. Kasturi, D. Mihalcik, H. Li, D. Doermann, and T. Drayer, *Performance Evaluation of Object Detection Algorithms III*, Quebec City, Aug. 2002, 965–969.

Universal Document Management System for the Mobile Warrior

H. Alam R. Hartono F. Rahman¹ Y. Tarnikova T. Tjahjadi C. Wilcox

BCL Technologies Inc., 990 Linden Dr, Suite #203,
Santa Clara CA 95050, U.S.A.
fuad@bcltechnologies.com

Abstract

A mobile warrior, with respect to US Army philosophy and hierarchy, is someone who has complete access to any document and information in a mobile environment related to his/her current assignment. This implies that a mobile warrior will be able to access up-to-date task-specifications, modifications in assignment goals, documentation of any electronic or non-electronic gears including arsenal he/she is using, access to basic maintenance literature and emergency medical and other procedures, including catastrophic incidences such as a chemical weapons attack. Above all, a mobile warrior is always connected electronically to the chain of command. This presentation outlines the research being carried out at BCL Technologies Inc. under a Small Business Innovation Research (SBIR) contract (currently in Phase II) in close supervision of the United States Army Communication Electronics Command (CECOM). The overall objective of this research is to combine multiple key technologies in the area of document and web engineering to enable a mobile warrior to have universal access to information in terms of a global document management system.

1. Introduction

BCL Knowledge Management Center (KMC) will offer universal information access to the Mobile Warrior. Principal aims of this system will be to allow an “agile commander” on the move to have total control over the command structure, to allow individual US Army personnel to have access to critical and time-sensitive information and data, and to allow battle planners to analyze and direct battles using the knowledge base. Based on the research carried out in Phase I and undergoing consultation with CECOM about the practical aspects of such technology deployment, Phase II is concentrating on the following aspects of active research:

- Design of a Global Portal with Intelligent Information Flow: A portal is being designed that provides secured access to the KMC via different types of connections using XML/XSLT style sheets. Functionalities researched include an intelligent Data Stream Controller that will push different levels of data depending on the speed characteristics, requested information and security clearance of connected devices. On low speed characteristics, summarization of the data is considered.
- Intelligent Search: Performing intelligent search of documents in the KMC using natural language complex queries.
- Design of an Intelligent Document Server: A document server is being researched to store and serve documents/data/information to any type of connected device.
- Design Tools for Battlefield Planning: BCL is working on developing a set of tools that will allow commanders to access the KMC for analysis, visualization and interpretation of the data. This will include graphical interfaces (GUIs), tools for battle planning, simulation, troop reassignment and movement, risk analysis and other battlefield planning activities.

We are currently working on developing a suite of technologies related to these Command and Control (C2) applications for such a mobile warrior. The specific technology directly related to and developed in this ongoing project involves the following:

- Document Conversion: Creation of a common data repository where any type of document can be stored. The core technology of this repository is an innovative any-to-any document conversion engine. This document conversion has to guarantee preservation of the look-and-feel of the document, in addition to being highly accurate in content retention and reproduction. The target formats include HTML and XML, in order to be able to use web-based access protocols.

¹ Corresponding author

- XML Representation: Accessing stored data from anywhere using any device. This involves efficient XML representation of any document to allow access to the data using any type of networked device.
- Web Page Re-authoring: To be able to browse any web document using any device (PDAs, cell phones etc.) for any site. This involves significant automated web page re-authoring.
- Document Summarization: To create a concise and accurate summary of documents. This is specially suited for low speed small display devices.
- Document Classification: Since it is possible to convert all documents to a common HTML/XML representation, it is required to classify these web documents in a set of pre-defined categories. This classification is often a pre-processing stage in many other applications.

This paper discusses how we are using these diverse technologies within a unified framework. Also included will be the challenges we are facing and how other researchers can benefit from solutions we have developed.

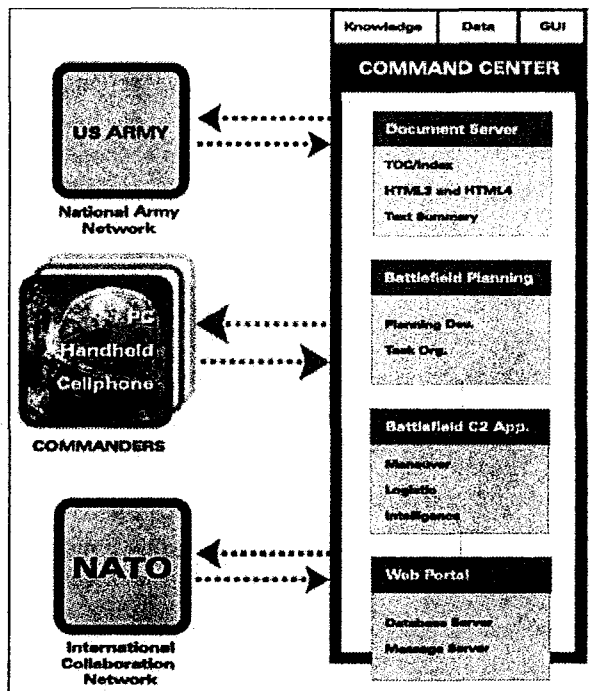


Figure 1: BCL Knowledge Management Center (KMC)

2. BCL Knowledge Management Center (KMC)

Figure 1 shows a high level schematic of BCL Knowledge Management Center (KMC). During Phase I research, BCL continually updated its understanding of the proposed technology in terms of applicability and relevance within the US Army. It is clear to us that for a practical knowledge based system the data has to be searchable with complex queries, a variety of devices

have to be accommodated, device specific XML/XSLT routing of content has to be incorporated and adaptable networking has to be supported to cope with a range of network speeds. In Phase I, BCL demonstrated the proof-of-concept of specific components of a system for Universal Information Access for the Mobile Warrior in terms of algorithms and application. The technology areas included design of a document repository, XML conversion systems, parsing, classification, segmentation, contextual analysis, segment labeling, summarization, segment sequencing and mobile universal access. BCL's technology aims to automatically convert documents into XML formats and store them in an XML based metadata database. As a natural progression to this, Phase II will include generating XSLT style sheets for various network devices for the most efficient intelligent routing of information. This will aim to develop an adaptive network where device types and network speed is taken into consideration while routing information to specific requests.

3. Technology Developed in Phase I

In Phase I, BCL proposed to research a suite of technologies related to C2 applications for a mobile warrior. Here we discuss the technology developed based on Phase I research. BCL principally developed the following viable technologies:

- Creation of a common data repository where any type of document can be stored. The core technology of this repository was an innovative any-to-any document conversion engine. The target formats included HTML and XML.
- Accessing the stored data from anywhere using any device. XML representation allows us to access the data using any type of networked device.
- Universal web browsing using any device for any site.
- Unique document summarization technology. This is specially suited for low speed small display devices.

There was also peripheral research done in other areas such as document classification and multiple device access that support these innovative technologies. In the rest of this section, a brief discussion of the technology researched in Phase I is presented.

3.1 Design of a Document Repository

The aim here is to store any type of document in a versatile XML format. The XML formatting then allows users to access the documents using any device and any network speed. XML not only provides information about the content of the documents, it also provides information about the structure and type of the data, allowing automatic selection of transmission

modes. For example, if a low speed low display device requests a document, it is possible to automatically switch to a summarized version of the document. More information on the summarization techniques innovated by BCL is described later. The BCL document repository consists of 3 sub-systems: input sub-system, storage sub-system and display sub-system (Figure 2). This allows automatic document submission and viewing via the web using various handheld and other devices.

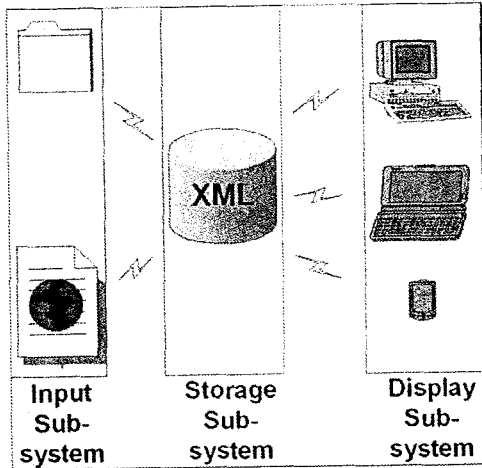


Figure 2: Document Repository

3.1.1 Input Sub-System

Figure 3 shows the principal stages of the input sub-system. Any static document submitted to the repository is automatically converted to PDF format. BCL has researched document conversion into XML from PDF format in depth. This is designed as an XML server solution that can accept documents for conversion through the standard web browser. We can currently convert PDF (Portable Document Format), RTF (Rich Text Format), TXT (Simple Text Format), DOC (Microsoft Word Format), XLS (Microsoft Excel Format), and PPT (Microsoft PowerPoint Format) files into XML.

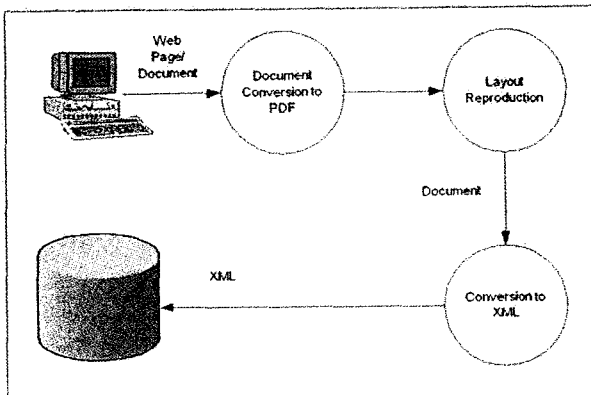


Figure 3: Input Sub-System

3.1.2 Storage Sub-System

The documents are automatically converted to XML and kept in an XML based metadata database. The system then adds the appropriate logical label tags and XML/XSLT based style sheets that allow the documents to be displayed on whatever display device the client chooses, all while faithfully preserving the original content and flow. The server application then indexes and catalogues the documents for the user, ready for access by the PDA or other handheld/desktop device. Once it is available in XML format, our automatic summarization modules generate a summary of the document [1]. This is also stored in an XML format.

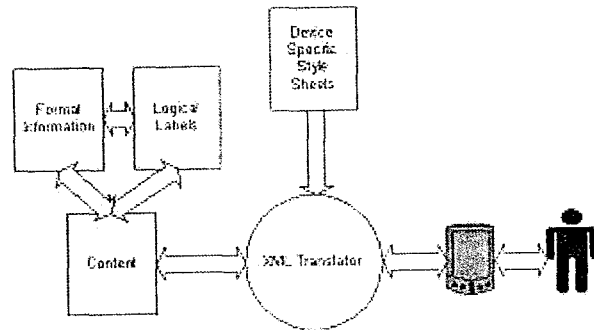


Figure 4: Display sub-system

3.1.3 Display Sub-System

It is possible to have three different versions of the same document in the repository: the original document that can be accessed and downloaded using the standard interface, the converted XML document, that can be accessed using a standard web interface, and the converted XML summary of the document that can be easily accessible via handheld devices such as a PDA [2] (Figure 4).

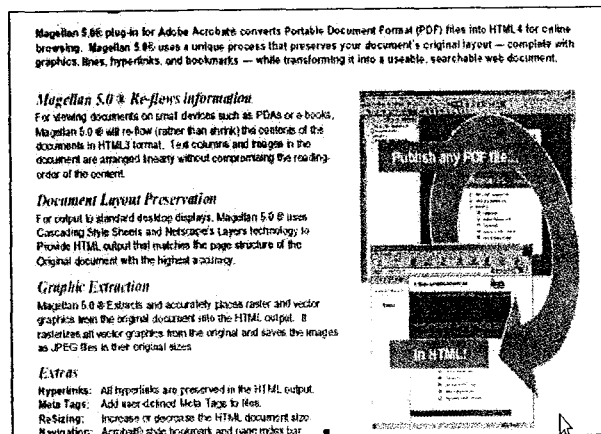


Figure 5: An example PDF document

3.2 Building a Web Based Document Conversion Scheme

BCL has demonstrated the proof-of-concept of a multi-format document conversion scheme. BCL demonstrated the proof-of-concept of technology that allows all these different formats to be ported onto the mobile platform by allowing XML translation. Figure 5 shows an example of a PDF document. Figure 6 shows how this document can be converted to XML while maintaining the layout, look and feel of the original document using BCL's technology.

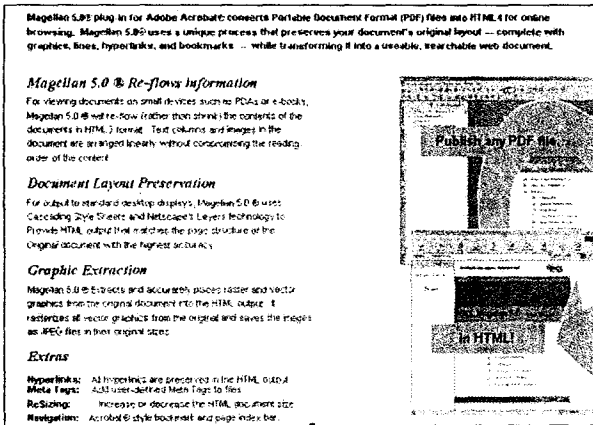


Figure 6: Converted to XML

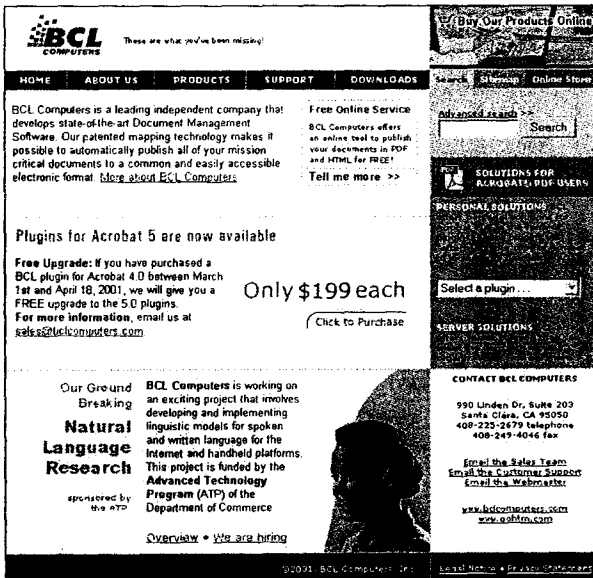


Figure 7: An XML document

3.3 Building a Robust Parser

BCL successfully researched a robust parser for parsing XML files and tested it extensively. Since BCL's technology allows all documents to be translatable into XML, this technology is invaluable in performing the next levels of analysis.

3.4 Classification of Documents

Classifying documents into various pre-defined groups is a requirement for high accuracy document processing and information extraction systems. BCL has successfully demonstrated the proof-of-concept of techniques for classifying documents into various pre-defined classes using Support Vector Machines (SVMs) [3]. Since all documents are now translatable onto the mobile platform in XML format, this technology focuses on classifying these types of documents. We used this hybridization of heuristics and SVMs in order to increase the efficiency of the overall system.

3.5 Document Segmentation

BCL has successfully demonstrated the proof-of-concept of the technology to segment documents into logical zones. Since XML is largely the principal format for the mobile platform, this technology focuses on the analysis of the structure of such documents and is achieved by exploiting the XML data structure.

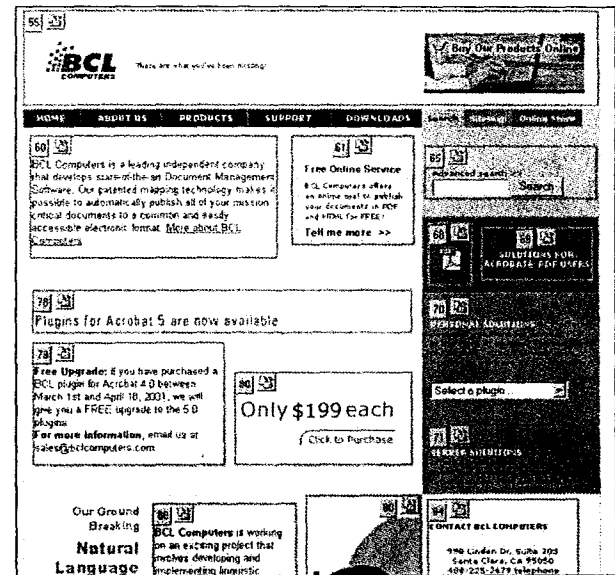


Figure 8: Partial results of segmentation

Figure 7 shows a multi-column document. Figure 8 shows partial results of the segmentation of this document using the segmentation algorithms.

3.6 Contextual Analysis and Segment Labeling

Phase I research has resulted in the demonstration of the proof-of-concept of the technology that can analyze and classify each constituent segment for its context based on the type and size of content using a rule-based engine [4]. This involves parameterizing the labels by assigning quantifiable weights.

Figure 9 shows the results of this process on a XML document. In this example, the segment in question is

assigned a processing type of "Structure" and a segment classification of "Large Story". This information is invaluable for secondary processing, such as producing a summary.

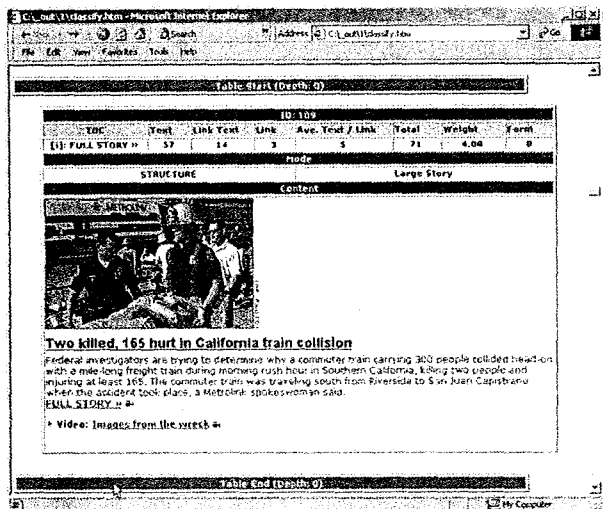


Figure 9: An illustration of contextual analysis and labeling

3.7 Summarization

Summarization is a key technology associated with universal information access for the mobile warrior. Since the display area and processing power of all handheld devices is considerably lower than a conventional desktop computer, summarizing content will be a necessary component in making information available to the mobile force quickly and efficiently [5]. Phase I research actively explored and demonstrated the proof-of-concept of the technology to summarize traditional documents (such as MS word, or PDF) and XML based directional documents based on the type and size of the content using a rule-based engine.

- [\(FORM\)SOLUTIONS FOR ACROBAT® PDF USERS](#)
- [BCL Computers is working on](#)
- [BCL Computers is a leading](#)
- [Free Upgrade: If you have](#)
- [CONTACT BCL COMPUTERS](#)
- [sponsored by](#)
- [Drake \(PDF to RTF\)](#)
- [Server Solutions](#)
- [Acrobat Plugins](#)
- [Legal Notice • Privacy Statement](#)
- [\(FORM\)Advanced search >>](#)

Figure 10: Tabular summary (TOC)

3.7.1 Tabular Summary

The document is summarized based on its structure. This summary is used to put together a Table of

Contents (TOC) that will act as both a summary and a navigation tool for small handheld displays. For example, the document segmented and analyzed in Figure 8 can now be summarized as shown in Figure 10. >From the display of Figure 10, if the first entry is selected, the display of Figure 11 is obtained. In the same fashion, Figure 12 gives the content associated with the labels "BCL Computers is working on".

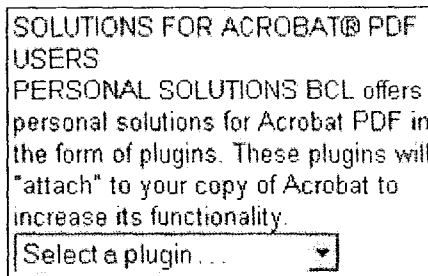


Figure 11: The content



Figure 12: Detailed Content

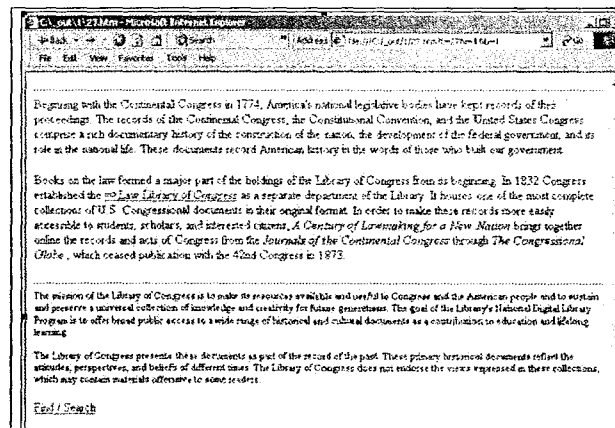


Figure 13: An HTML/XML document

Beginning with the Continental Congress in 1774, America's national legislative bodies have kept records of their proceedings. Books on the law formed a major part of the holdings of the Library of Congress from its beginning.

Figure 14: An NL summary

3.7.2 Content Summary

The document is summarized using Natural Language Processing (NLP) techniques that analyze the content of the document from a linguistic point of view and generate a concise summary. For example, Figure 13 shows an HTML/XML document. Generating a TOC based summary of this segment is not ideal, as the textual content of this segment can never be adequately expressed using a single line summary. In these cases, natural language processing techniques have been used by BCL to generate a summary that is representative of the text in the original document. Figure 14 shows a natural language summary of the document shown in Figure 13. BCL's summarization technology has been tested heavily under realistic conditions and has been shown to be very effective.

3.8 Segment Sequencing

Summarizing a document raises the question of re-purposing a document based on the importance of the content of the segments. Since the original document is segmented into smaller sub-documents, these need to be re-purposed by allowing more important content to appear earlier in the display resulting in more efficient and intelligent displays. In the time of writing this Phase II proposal, research is still continuing on this topic. BCL plans to use three different methods to achieve this:

- Content Weight: In many cases, it is quite accurate to assume that the "importance" of content will be directly related to its size. This can be quantified using the concept of content weight. Figure 10 shows the order in which the labels are ordered.
- TOC Weight: BCL also plans to use TOC weight to select the order as estimated by the labeling algorithms. For example, Figure 15 associates with labels "Server Solutions", and it is placed towards the bottom of the TOC. It is also noted that labels with forms associated with their content are shown as part of their label. For example, the label "(FORM) Advanced Search" is associated with segment 65 (Figure 8). Figure 16 shows the content associated with this label. For illustrative purposes, this is only done with forms here, but it is entirely possible to associate other classes such as "main text", "images", "navigation" etc with the labels. This explicit labeling often helps in guiding the reader towards the correct type of content and makes locating information on the web easier. Finally, content from the end of the page is placed as close to the end as possible (Figure 17).

3.9 Device Independence

BCL has shown that its technology enables multiple device access to data and information, especially handheld devices. The proof of concept is demonstrated on handhelds such as Palm, Pocket PC, Jornada, etc.

and cellular phones using WAP, iMode (NTT DoCoMo), J-Sky (J-Phone) and EZweb (KDDI) formats, in addition to standard desktop and laptop computers. This is made possible because BCL's technology can use XML/XSLT style sheets for various devices. These style sheets contain device specific information, functional capabilities, supportable network speeds and other specifications. This information is used while deciding how to handle specific devices connected to the network.

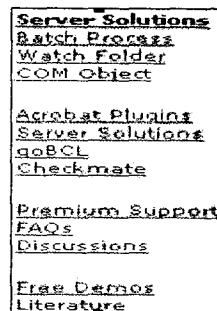


Figure 15: Content

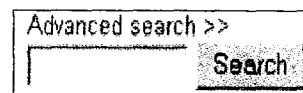


Figure 16: Content with form



Figure 17: Detailed Content

4. Future Challenges

We have started working on the Phase II of the project since February of this year. So far we have concentrated on developing prototypes of a KMC using the hardened user requirements as requested by the US Army. Many of the challenges that are in front of us include:

- Detection of network speed: Since we need to know how much information can a device connected to the network can handle at all times, we need to be able to detect the network speed accurately.
- Detection of Device Types: Since a variety of small-screen display devices can be connected to KMC, we need to know what types of devices are connected and what type of display capabilities they possess.
- Mobile Security: It is very important to ensure absolute security. We are actively looking to ensure that the KMC offers the state of the art data encryption and security.
- Document Integrity: Converting a document incorrectly may lead to loss of integrity. We are spending a lot of effort in trying to ensure that

document integrity is maintained.

- Universal Document summarization: Since low speed, small-screen devices may use summarized version of documents, we are actively researching to build an accurate summarizer. This summarizer will work on any type of document, not only XML documents.
- Intelligent Search: BCL is actively researching a natural language based contextual search engine. This will be based on BCL's expertise in natural language processing.
- Planning Tools: Having access to information on demand is not very useful if that information/data cannot be viewed and analyzed. BCL is working on developing a set of tools to allow commanders on the move to effectively visualize and analyze the collected information.

5. Conclusions

This presentation has outlined some of the exciting new technologies that are being researched at BCL Technologies as part of the KMC project. The KMC offers an integration of state of the art document understanding, analysis, and conversion techniques within a common operating framework. BCL plans to deploy a prototype KMC system for CECOM and extensively exploit the test environment to refine the system to the point where it can be adopted as a backbone for the C2 and planning activities of the U.S. Army in the next two years.

Acknowledgement

Our thanks to the US Army Communication Electronics Command (CECOM) for supervising the Small Business Innovation Research (SBIR) grant that funded this research.

References

- [1] F. Rahman, H. Alam and R. Hartono. Understanding the Flow of Content in Summarizing HTML Documents. Int. Workshop on Document Layout Interpretation and its Applications, DLIA01, 2001.
- [2] F. Rahman, H. Alam, R. Hartono and K. Ariyoshi. Automatic Summarization of Web Content to Smaller Display Devices. 6th Int. Conf. On Document Analysis and Recognition, ICDAR01, pp. 1064-1068, 2001.
- [3] F. Rahman, Y. Tarnikova and H. Alam. Exploring a Hybrid of Support Vector Machines (SVMs) and a Heuristic Based System in Classifying Web Pages. Document Recognition and Retrieval X, 15th Annual IS&S/SPIE Symposium, pp. 120-127, 2003.
- [4] F. Rahman, H. Alam and R. Hartono. Content Extraction from HTML Documents. Int.

Workshop on Web Document Analysis, WDA01, pp. 7-10, 2001.

- [5] F. Rahman and H. Alam. Challenges in Web Document Summarization: Some Myths and Reality. Document Recognition and Retrieval IX, Electronic Imaging Conference, SPIE 4670-27, 2002.

Demos & Abstracts

The Gamera framework for building custom recognition systems

Michael Droettboom Karl MacMillan Ichiro Fujinaga

Digital Knowledge Center, Sheridan Libraries

The Johns Hopkins University

3400 N. Charles St., Baltimore, MD 21218

mdboom@jhu.edu, karlmac@jhu.edu, ich@music.mcgill.ca

Abstract

This paper describes the Gamera framework for building custom document recognition systems. This open-source system is designed to support the test-and-refine development cycle: an important style for developing recognition systems that work with difficult historical documents, since the solutions are often non-obvious. This paper explains the overall architecture of the system, in addition to detailed information on recent research subprojects and their performance on real-world data.

1 Introduction

Gamera is a framework for the creation of structured document analysis applications by domain experts. Domain experts are individuals who have a strong knowledge of the documents in a collection, but may not have a formal technical background. The goal is to create a tool that leverages their knowledge of the target documents to create custom applications rather than attempting to meet diverse requirements with a monolithic application.

We intend to use Gamera to develop applications for a number of diverse collections, including medieval manuscripts, lute tablature, Greek text, and handwritten text. There are many collections that would benefit from this technology and would represent a diverse range of document analysis challenges [1].

This paper gives an overview of the architecture and design principles of Gamera. In addition, recent additions to Gamera are discussed in more detail, with an analysis of their performance on real-world data.

2 Architecture overview

Developing recognition systems for difficult historical documents requires experimentation since the solution is often non-obvious. Therefore, Gamera's primary goal is to support an efficient test-and-refine development cycle. Virtually every implementation

detail is driven by this goal. For instance, Python [2] was chosen as the core language because of its introspection capabilities, dynamic typing and ease of use. It has been used as a first programming language with considerable success [3]. C++ is used to write plugins where runtime performance is a priority, but even in that case, the Gamera plugin system (Section 2.3.2) is designed to make writing extensions as easy as possible. Gamera includes a full-fledged graphical user interface that provides a number of shortcuts for training, as well as inspection of the results of algorithms at every step. By improving the ease of experimentation, we hope to put the power to develop recognition systems with those who understand the documents best. We expect at least two kinds of developers to work with the system: those with a technical background adding algorithms to the system, and those working on the higher-level aggregation of those pieces. It is important to note this distinction, since those groups represent different skill sets and requirements.

In addition to its support of test-and-refine development, Gamera also has several other advantages that are important to large-scale digitization projects in general. These are:

- Open source code and standards-compliance so that the software can interact well with other parts of a digitization framework
- Platform independence, running on a variety of operating systems including Linux (Figures 2–5) and Microsoft Windows (Figures 8–9)
- A workflow system to combine high-level tasks
- Batch processing
- A unit-testing framework to ensure correctness and avoid regression
- Rich user interface components for development and classifier training

- Recognition confidence output so that collection managers can easily target documents that need correction or different recognition strategies

2.1 Tasks

Gamera has a modular plugin architecture. These modules typically perform one of five document recognition tasks:

1. Pre-processing
2. Document segmentation and analysis
3. Symbol segmentation and classification
4. Syntactical or structural analysis
5. Output

Each of these tasks can be arbitrarily complex, involve multiple strategies or modules, or be removed entirely depending on the specific recognition problem at hand. The actual steps that make up a complete recognition system are completely controlled by the user.

2.1.1 Pre-processing

Preprocessing involves standard image-processing operations such as noise removal, blurring, deskewing, contrast adjustment, sharpening, binarization, and morphology. Close attention to and refinement of these steps is particularly important when working with degraded historical documents.

2.1.2 Document segmentation and analysis

Before the symbols of a document can be classified, an analysis of the overall structure of the document may be necessary. The purpose of this step is to analyze the structure of the document, segment it into sections, and perhaps identify and remove elements. For example, in the case of music recognition, it is necessary to identify and remove the staff lines in order to be able to properly separate the individual symbols. Similarly, text documents may require the removal of figures.

2.1.3 Symbol segmentation and classification

The segmentation, feature extraction, and classification of symbols is the core of the Gamera system. The system allows different classifiers to be plugged-in. These classifiers come in two flavors: “interactive” classifiers, where examples can be added and the results tested immediately, and “non-interactive” classifiers, that are highly optimized but static. Gamera has a generic and flexible

XML-based file format (Section 2.1.5) to store classifier data, but for efficiency, the “non-interactive” classifiers can also define their own format containing Pre-parsed or pre-optimized data. At present, we have an implementation of the k -nearest neighbor (k -NN) algorithm [4] whose weights are optimized using a genetic algorithm (GA) [5]. We have tested the extensibility of our classifier framework by porting a simple back-propagating neural network library to Gamera [6]. We also plan to examine what modifications would be necessary to support stateful classifiers, such as hidden Markov models.

2.1.4 Syntactical or structural analysis

This process reconstructs a document into a semantic representation of the individual symbols. Examples of this include combining stems, flags, and noteheads into musical objects, or grouping words and numbers into lines, paragraphs, columns etc. Obviously, this process is entirely dependent on the type of document being processed and is a likely place for large customizations by knowledgeable users.

Section 3.1 describes a pattern matching engine designed to help with many of these analysis problems.

2.1.5 Output

Gamera stores groups of glyphs in an XML-compliant format (Figure 1). This makes it very easy to save, load, and merge sets of training data. Since a run-length encoded copy of the glyph is included, it is easy to load the original images and inspect, edit or generate new features from them.

Output of data that is specific to a particular type of document, i.e. post-structural interpretation, is deliberately left open-ended, since different domains will have different requirements. For example, the GUIDO [7] file format is used for symbolic music representation by our Gamera-based music recognition application.

2.2 Graphical user interface

Since document recognition is an inherently visual problem, a graphical user interface (GUI) is included to allow the application developer to experiment with different recognition strategies. At the core of the interface is the console window (Figure 2) which allows the programmer to run code interactively and control the system either by typing commands or using menus. All commands are recorded in a history, which can later be used for building automatic batch scripts. The interface also includes a simple image viewer, image analysis tools (Figure 3), and a training interface for the learning classifiers (Figure

Figure 1: An example glyph from the Gamera XML file format.

```

<?xml version="1.0" encoding="utf-8"?>
<gamera-database version="2.0">
  <glyphs>
    <glyph uly="798" ulx="784" nrows="15" ncols="12">
      <ids state="MANUAL">
        <id name="lower.c" confidence="1.000000"/>
      </ids>
      <!-- Run-length encoded binary image (white first) -->
      <data>
        5 6 4 9 2 4 2 4 2 4 3 3 1 4 6 5 8 4 9 3 8 4 9 4 8 5 7 5 3 3 3 9 3 9 6
        4 2 0
      </data>
      <features scaling="1.0">
        <feature name="area">
          180.0
        </feature>
        <feature name="aspect_ratio">
          0.8
        </feature>
        <feature name="compactness">
          0.584269662921
        </feature>
        <feature name="moments">
          0.219907407407 0.2288888888889 0.0697385116598 0.1266111111111
          0.0505606995885 0.0203254388586 0.0177776861746 0.00727370913662
          0.0488995911061
        </feature>
      </features>
    </glyph>
  </glyphs>
</gamera-database>

```

4). The training interface allows the user to create databases of labeled glyphs, including through the merging and splitting connected components.

The interface can easily be extended to include new elements as modules are added to Gamera. The entire GUI is written in Python, using the wxPython toolkit [8].

2.3 Implementation details

2.3.1 Image classes

The storage and manipulation of images is one of the most important aspects of Gamera. Gamera must provide not only general-purpose image manipulation functions, but also infrastructure to support the symbol segmentation and analysis. The features of the Image classes in Gamera include:

Polymorphic image types. Gamera images can be stored using a number of different pixel types, including color (24-bit RGB), greyscale (8- and 16-bit), floating point (32-bit) and bi-level images, though new image types can be added as desired.

Consistent programming interface. The interface to all types of images (in both C++ and Python) is the same. While some methods are not available for all image types (e.g. thresholding a bi-level image would not make sense), in many cases image types are interchangeable, meaning types can be changed at different points in the development process.

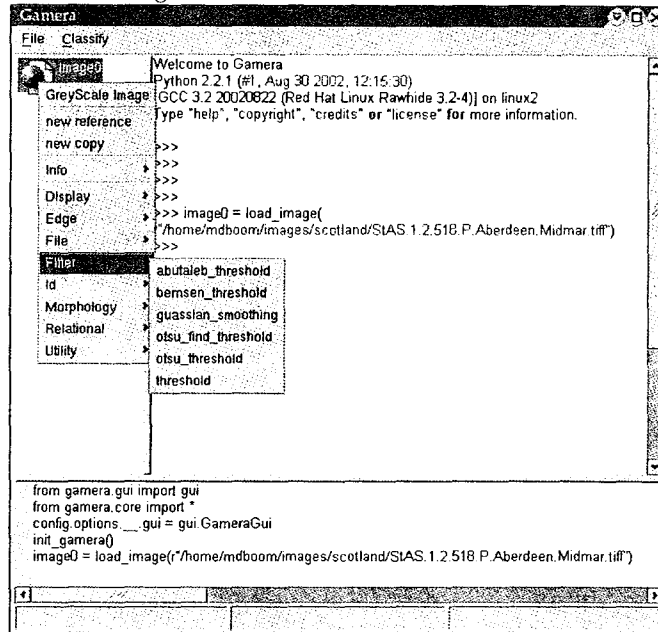
Use of existing code libraries. The image classes have been designed especially to make transferring code from other C/C++ image processing libraries as easy as possible. For example, many algorithms in the VIGRA library [9] can be used in Gamera without modification. Using C libraries, such as XITE [10], often requires only a few minor modifications of the code. This ability has reduced our development time considerably.

Portions of images. The image classes allow for the flexible and efficient representation of portions of images, including non-rectangular regions, without resorting to memory copying. The bi-level images actually store 16-bits-per-pixel, so that labeling information can be stored to define connected components. This uses a considerably smaller memory footprint than using separate data for each connected component.

2.3.2 The plugin system

Writing wrapper code to call C/C++ from Python is a time-consuming, error-prone and repetitive task. A number of general-purpose tools exist to help automate this process, including SWIG (<http://www.swig.org>) and Boost Python (<http://www.boost.org>). In fact, an earlier version of Gamera used Boost Python, but the additional function-call overhead for our highly polymorphic image types lead to poor performance of that system as a whole. We have since developed our own

Figure 2: The Gamera console window.



wrapper-generating mechanism specific to Gamera and its classes. This allowed us to provide optimizations and conveniences to the programmer that would not be possible with a more general approach.

To add a plugin function to Gamera, a programmer writes metadata about the function (in Python) and a single function to perform an image-processing task (in C++). Plugin functions are grouped into standard Python modules, which are collections of related classes and functions.

As an example, the metadata for the Morphology module (`morphology.py`) is listed below.

```
class erode_dilate(PluginFunction):
    self_type = ImageType([ONEBIT, GREYSCALE, FLOAT])
    args = Args([
        Int('times',
            range=(0, 10), default=1),
        Choice('direction',
            ['dilate', 'erode']),
        Choice('window_shape',
            ['rectangular', 'octagonal'])
    ])
...
class MorphologyModule(PluginModule):
    cpp_headers = ["morphology.hpp"]
    cpp_namespaces = ["Gamera"]
    category = "Morphology"
    functions = [erode_dilate, erode, dilate,
                rank, mean]
    author = "Michael_Droettboom_and_Karl_MacMillan"
    url = "http://gamera.dkc.jhu.edu/"
```

Each plugin function has a class that inherits from `PluginFunction`. There are a number of (static) members of this class that can be used to customize the function. `self_type` lists the types of images

that the function can be performed on. This information is important, since we must map the polymorphic method calls made in the Python environment to a particular compiled instance of the function in C++. The `args` member gives details about the function's arguments. This detail goes beyond what could be obtained automatically from a C++ function declaration to include elements such as ranges and named enumerations. This information is used both to automatically generate the function wrapper code and to generate dialog boxes within the GUI (Figure 5). At the module level, there is a single class to describe the entire module that inherits from `PluginModule`. This is used to specify which C++ headers or libraries need to be included, authorship information, and other miscellaneous hints to the C++ compiler. The plugin functions themselves are just free C++ functions. The declaration of the function for `erode_dilate` (in `morphology.hpp`) is given below. (The details of the algorithm itself are not important to understanding the plugin system.)

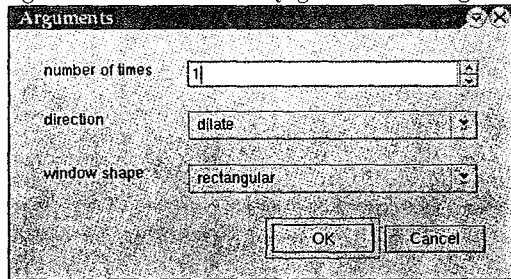
```
template<class T>
void erode_dilate(T&m, unsigned int times,
                 int direction, int window_shape) {
...
}
```

The Gamera build system uses the above metadata to generate a wrapper for the function. Since the function is templated, a different instance

Figure 3: The Gamera image viewer.



Figure 5: An automatically generated dialog box.



of the function is compiled for each of the image types it supports. The plugin functions are added to the Gamera Image class at runtime. This makes it possible to use standard dot syntax to call these methods. For example, the plugin function `erode_dilate` could be performed on `img0` with the statement `img0.erode_dilate(0, 0, 0)` (instead of `erode_dilate(img0, 0, 0, 0)`).

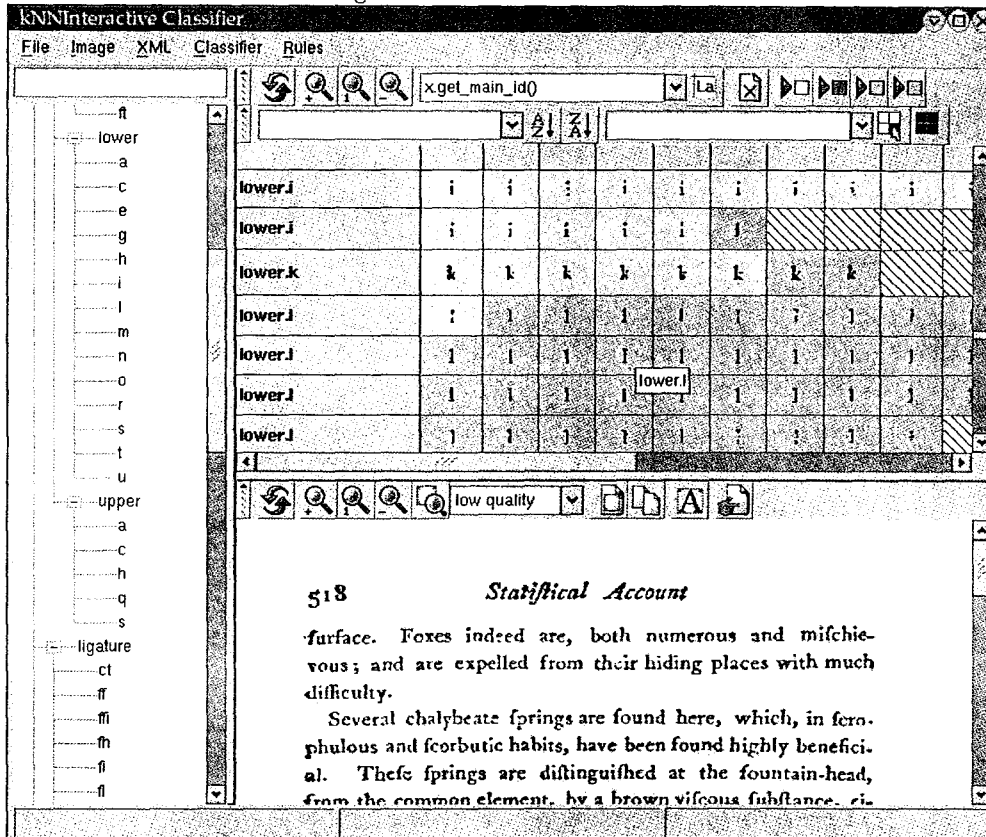
2.3.3 Workflow

Gamera has a workflow infrastructure that allows the end user to easily tie tasks together into a complete recognition system. The workflow engine handles the persistence of data at each step, so that if a particular step fails, it is possible to roll-back and begin in the middle of a process without repeating preceding steps. Again, this encourages the test-and-refine development model. We plan to extend this workflow concept to include a simple visual programming language (signal-flow graph) [11], to make it easier for non-programmers to customize and build their own systems.

2.3.4 Unit-testing

Gamera includes a convenient unit-testing framework for testing plugins or other functions. If no unit tests are written for a given function, stock source images are used with default arguments, and the resulting image is compared to one that has been declared correct. If a step requires more rigorous testing, a custom unit test can be written. In either case, Gamera handles all of the details of saving and

Figure 4: The classifier window.



loading images to disk and verification.

3 Recent developments

That completes the overview of the architecture of Gamera. Some of our recent research includes designed a declarative pattern matching engine, improving the handling of degraded documents, and using clustering techniques to reduce manual training times and provide additional verification. These three subprojects are discussed in detail below.

3.1 Pattern matching engine

A pattern matching engine is included to assist with basic structural analysis problems. Using a logic programming language such as Prolog [12], or perhaps a domain-specific minilanguage, would be suitable for this purpose. However, we felt it would decrease the usability of the system to add yet another programming language to the mix. Therefore, the pattern matching engine is designed so that authors simply write specialized Python functions that add some declarative programming to what is still primarily an imperative approach.

An example rule function for finding periods at the end of sentences is given below.

```
# Find periods at the end of sentences
# (a) function signature
def find_periods(a="dot", b="letter*"):

    # (b) expression
    if (Fudge(a.lr.y) == b.lr.y and
        a.ul.x > b.lr.x and
        a.ul.x - b.lr.x < 20):

        # (c) operation
        period = a.copy()
        period.classify_heuristic(
            'punctuation.period')

        # (d) added, removed
        return [period], []
```

The argument signature (a) defines which symbols will be applied to the function, using regular expressions to match the class name. (Remember, the symbols have already been classified in the previous stage.) The regular expression syntax used is designed to be as convenient and familiar as possible, and is based on what one might find in a Unix shell. In the example, the variable a will be unified with glyphs classified as dot, and b with glyphs of any class beginning with letter. The body of the

function generally contains an expression (b) and an operation (c). The expression tests for some criteria, such as the relative position of glyphs. The operation will then produce data to make note of the found pattern. Lastly, the function returns two lists of glyphs (d) that should, respectively, be added or removed from the global set of glyphs. If the function causes no direct side-effects (i.e. doesn't directly modify data, but instead returns modified copies), the Gamera GUI is able to provide undo functionality on top of the pattern matching engine, which helps to support the test-and-refine development model.

The time-complexity of the pattern matching engine is improved by storing all the glyphs in a grid index. Only groups of glyphs in the same or adjacent cells are used as candidate groups for arguments to a pattern matching function. The size of the grid cells can be adjusted, to change the amount of adjacency possible within a pattern.

The pattern matching engine has proven to be useful for solving a number of problems, including finding punctuation and grouping parts of cursive words. We plan to improve it to support global optimizations and pattern-ordering facilities. There are also improvements in runtime that could be achieved by indexing the functions by their expressions. Since the approach is so straightforward, however, it is difficult for the pattern matching engine to achieve optimal runtime efficiency. In any case, it remains a very useful rapid prototyping tool, again supporting the test-and-refine model. In addition, the full power of Python can always be used to perform syntactic or structural analysis when the pattern matching tools are too weak or inefficient.

3.2 Character degradation

Most commercial optical character recognition (OCR) systems are designed for well-formed, modern business documents. Recognizing older documents with low-quality or degraded printing is more challenging, due to the high occurrence of broken and touching characters. Gamera includes unique approaches for dealing with both of these problems. These algorithms have been very successful on a set of real-world historical documents.

For the purposes of this paper, a *connected component* (CC) is a set of black pixels that are 8-connected. By definition, characters are *broken* when they are made up of too many CCs, and they are *touching* when they are made up of too few CCs. Broken characters can not be joined simply by the distance of the CCs alone, since two intentionally separate characters can often be closer than the two parts of an accidentally broken character.

3.2.1 Other approaches

Thresholding converts a color or greyscale image to a bi-level image, such that black is used to indicate the presence of ink on the page and white is used to indicate its absence. Improving thresholding by looking for shades of grey in the areas where CCs almost touch, using entropy, can reduce the number of broken characters [13]. However, in many historical documents, the characters are completely broken on the page and intelligent thresholding, since it has no knowledge of the shapes of the target characters, performs poorly.

Active contour models (ACM), or snakes [14], find a vector outline for each symbol using certain constraints on the elasticity of the outline. Unfortunately, ACMs, which were designed for gross shape recognition, perform poorly on the fine details that are required to recognize printed characters.

Post-processing using some kind of language model, including a dictionary or n -grams of a language [15] has also been used to improve the recognition of broken and touching characters. However, such models are less useful for documents containing ancient languages, mixed-languages or a high occurrence of proper nouns.

Therefore, an ideal solution would include knowledge of the individual symbols without requiring a language-specific model.

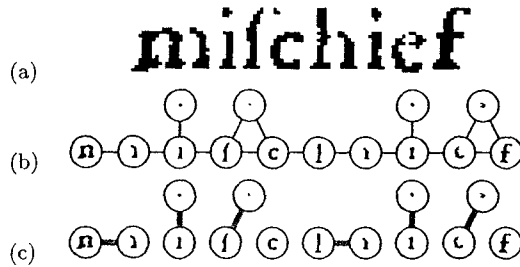
3.2.2 Broken character correction

The goal of the broken character connection (BCC) algorithm is to find an optimal way to join CCs on a given page that maximizes the mean confidence of all characters.

The algorithm begins by building an undirected graph in which each vertex represents a CC in the image. Two vertices are connected by an edge if the border of the bounding boxes are within a certain threshold of distance. Experiments demonstrated that this threshold is best set to $3/4$ of the average distance between all bounding boxes. (CCs can also be connected by morphological dilation, though the bounding box method is much faster and produces only slightly less accurate results.) This creates a forest of graphs where each graph is roughly equivalent to a word in the document. Figure 6(b) shows one such graph. A graph representation, rather than a string representation, of connectivity is necessary, since characters can be broken in the x - and/or y -direction and cycles can occur between CCs.

Next, all of the different ways in which the CCs can be joined are evaluated. Every possible connected subgraph is enumerated by performing a depth-first search from each vertex. The subgraphs created from the word in Figure 6(a) are shown in Table 1. To avoid enumerating duplicate possibili-

Figure 6: (a) an original image of a word from the testbed; (b) how the connected components are connected to form a graph; (c) the correct solution.



ties, the vertex v assigned a number N_v , and an edge is traversed from vertex a to b only if $N_b > N_a$. To improve runtimes, the depth of the search is limited to the maximum number of CCs that would typically make up a single broken character. This constant is adjusted automatically based on the amount of degradation in the image and is usually between 3 and 5. The “correctness” of each of the images represented by these subgraphs is evaluated by merging all of its CCs into a single image and sending it to the symbol classifier. The symbol classifier returns a confidence value that indicates how similar the merged image is to known symbols in the database.

Since we are using the k -nearest neighbor (k -NN) classifier [4] for symbol classification, it was most convenient to use a confidence measure based on distance. More elaborate ways of determining confidence, such as analyzing the clustering of symbols within the database, have been suggested, but they do not significantly affect the success of the BCC algorithm.

Once these subgraphs have been evaluated, an optimal combination of them must be found. Each subgraph is represented by a bit-string, in which a 1 indicates the presence of a vertex. The goal is to find combinations of subgraphs such that each vertex is used exactly once. By sorting the list of subgraphs in lexical order by their bitstrings, we can use the following criteria to do this efficiently:

Given subgraphs P and Q , Q can be added to P if it (a) does not have any vertices in common with P and (b) contains the lowest-numbered vertex that P is missing.

More formally,

Criterion A:

let V be the set of all vertices in the entire graph, and $P \subseteq V$, $Q \subseteq V$;

(a) $\forall v \in Q, v \notin P$;

(b) $\forall v \in V, N_v < \min(\forall w \in Q, N_w) \Rightarrow v \in P$.

The importance of sorting the list of subgraphs is that Q will always be after P in the list, thereby reducing the arity of the search tree. The algorithm starts by setting P to the empty set. All subgraphs Q that meet **Criterion A** are combined with P . The union of P and Q is then used as P recursively, until $P = V$. The mean confidence of all the Q 's used in each path of recursion is used to evaluate the combination. The combination with the highest mean confidence is chosen as the correct combination.

The runtime of the combination-finding can be improved by exploiting the fact that each subgraph P has a fixed set of subgraphs Q' that are contiguous in the sorted subgraph list and that meet **Criterion A(b)**. These sets can be found ahead of time in linear times relative to the number of vertices ($O(|V|)$). Once this is done, we've reduced the search space from each P considerably. For example, Table 1 shows the beginning and ending (non-inclusive) indices of Q' for each subgraph in the *from* and *to* columns. In this example, the mean size of Q' is 4.7, which corresponds the mean arity of the search tree. Without this optimization, the mean arity would be $\frac{|V|}{2} = 37$. As an additional optimization, “memoization”, borrowed from dynamic programming, is used to store the best result of all subtrees of the search so that identical subtrees do not need to be traversed multiple times.

While a full runtime complexity analysis of the entire BCC algorithm is beyond the scope of this paper, the asymptotic upper bound is $O(n \ln n)$, where n is the number of vertices in the graph. However, when there are no cycles, the runtime is reduced to roughly $O(kn)$, where k is the maximum size of the subgraphs (usually $3 \leq k \leq 5$).

3.2.3 Touching character correction

To deal with touching characters, we have an entirely different approach that was first used for optical music recognition [16]. The symbol classifier is trained with examples of touching characters, which are given a class name beginning with the special token `_split`. When the classifier later matches an unknown to an example starting with `_split`, it will perform a splitting operation on the connected component, and then recursively classify the results. The splitting operation can be any function that examines a connected component and splits it into multiple connected components through some kind of heuristic process.

For example, we have implemented primitive `splitx` function that splits a connected component by finding a minimum projection on the x -axis near the center of the connected component. If we let p_x be the size of the projection (number of black pixels)

Table 1: The bitfields and jumps created from the graph in Figure 6(b). The bits have been reversed to match the way the characters are read (left-to-right).

| # | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | from | to |
|----|---|---|---|---|---|---|---|---|---|---|----|----|----|----|------|----|
| 0 | | | | | | | | | | | | | | | 7 | 18 |
| 1 | | | | | | | | | | | | | | | 18 | 28 |
| 2 | | | | | | | | | | | | | | | 28 | 34 |
| 3 | | | | | | | | | | | | | | | 34 | 41 |
| 4 | | | | | | | | | | | | | | | 28 | 34 |
| 5 | | | | | | | | | | | | | | | 18 | 28 |
| 6 | | | | | | | | | | | | | | | 18 | 28 |
| 7 | | | | | | | | | | | | | | | 18 | 28 |
| 8 | | | | | | | | | | | | | | | 28 | 34 |
| 9 | | | | | | | | | | | | | | | 34 | 41 |
| 10 | | | | | | | | | | | | | | | 41 | 45 |
| 11 | | | | | | | | | | | | | | | 28 | 34 |
| 12 | | | | | | | | | | | | | | | 28 | 34 |
| 13 | | | | | | | | | | | | | | | 28 | 34 |
| 14 | | | | | | | | | | | | | | | 18 | 28 |
| 15 | | | | | | | | | | | | | | | 18 | 28 |
| 16 | | | | | | | | | | | | | | | 18 | 28 |
| 17 | | | | | | | | | | | | | | | 18 | 28 |
| 18 | | | | | | | | | | | | | | | 28 | 34 |
| 19 | | | | | | | | | | | | | | | 34 | 41 |
| 20 | | | | | | | | | | | | | | | 41 | 45 |
| 21 | | | | | | | | | | | | | | | 45 | 49 |
| 22 | | | | | | | | | | | | | | | 41 | 45 |
| 23 | | | | | | | | | | | | | | | 28 | 34 |
| 24 | | | | | | | | | | | | | | | 28 | 34 |
| 25 | | | | | | | | | | | | | | | 28 | 34 |
| 26 | | | | | | | | | | | | | | | 28 | 34 |
| 27 | | | | | | | | | | | | | | | 28 | 34 |
| 28 | | | | | | | | | | | | | | | 34 | 41 |
| 29 | | | | | | | | | | | | | | | 41 | 45 |
| 30 | | | | | | | | | | | | | | | 45 | 49 |
| 31 | | | | | | | | | | | | | | | 49 | 54 |
| 32 | | | | | | | | | | | | | | | 41 | 45 |
| 33 | | | | | | | | | | | | | | | 41 | 45 |
| 34 | | | | | | | | | | | | | | | 41 | 45 |
| 35 | | | | | | | | | | | | | | | 45 | 49 |
| 36 | | | | | | | | | | | | | | | 49 | 54 |
| 37 | | | | | | | | | | | | | | | 54 | 60 |
| 38 | | | | | | | | | | | | | | | 41 | 45 |
| 39 | | | | | | | | | | | | | | | 41 | 45 |
| 40 | | | | | | | | | | | | | | | 41 | 45 |
| 41 | | | | | | | | | | | | | | | 45 | 49 |
| 42 | | | | | | | | | | | | | | | 49 | 54 |
| 43 | | | | | | | | | | | | | | | 54 | 60 |
| 44 | | | | | | | | | | | | | | | 60 | 66 |
| 45 | | | | | | | | | | | | | | | 49 | 54 |
| 46 | | | | | | | | | | | | | | | 54 | 60 |
| 47 | | | | | | | | | | | | | | | 60 | 66 |
| 48 | | | | | | | | | | | | | | | 66 | 67 |
| 49 | | | | | | | | | | | | | | | 54 | 60 |
| 50 | | | | | | | | | | | | | | | 60 | 66 |
| 51 | | | | | | | | | | | | | | | 66 | 67 |
| 52 | | | | | | | | | | | | | | | 67 | 71 |
| 53 | | | | | | | | | | | | | | | 66 | 67 |
| 54 | | | | | | | | | | | | | | | 60 | 66 |
| 55 | | | | | | | | | | | | | | | 66 | 67 |
| 56 | | | | | | | | | | | | | | | 67 | 71 |
| 57 | | | | | | | | | | | | | | | 66 | 67 |
| 58 | | | | | | | | | | | | | | | 66 | 67 |
| 59 | | | | | | | | | | | | | | | 66 | 67 |

| # | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | from | to |
|----|---|---|---|---|---|---|---|---|---|---|----|----|----|----|------|----|
| 60 | | | | | | | | | | | | | | | 66 | 67 |
| 61 | | | | | | | | | | | | | | | 67 | 71 |
| 62 | | | | | | | | | | | | | | | 66 | 67 |
| 63 | | | | | | | | | | | | | | | 66 | 67 |
| 64 | | | | | | | | | | | | | | | 66 | 67 |
| 65 | | | | | | | | | | | | | | | 66 | 67 |
| 66 | | | | | | | | | | | | | | | 67 | 71 |
| 67 | | | | | | | | | | | | | | | 71 | 73 |
| 68 | | | | | | | | | | | | | | | 73 | 74 |
| 69 | | | | | | | | | | | | | | | 74 | 74 |
| 70 | | | | | | | | | | | | | | | 71 | 73 |
| 71 | | | | | | | | | | | | | | | 73 | 74 |
| 72 | | | | | | | | | | | | | | | 74 | 74 |
| 73 | | | | | | | | | | | | | | | 74 | 74 |

Figure 7: Example of touching characters being split using a projections-based split algorithm.



in column x , and d_x be the distance from the center along the x -axis ($|\frac{w}{2} - x|$), the score for each column is determined with the Euclidean distance equation:

$$s = \sqrt{p_x^2 + d_x^2} \quad (1)$$

The column with the minimum score is selected as the split point. Figure 7 shows an example of splitting a connected component, classified as `_split.splitx`, using this method. Of course, more sophisticated ways of splitting are possible. For example, we have also implemented an algorithm that removes vertical lines from compound musical structures in order to separate their components.

The runtime complexity of this approach is dependent on the underlying symbol classifier and the splitting operation. The splitting operation defined above runs in linear time.

3.2.4 Results

To evaluate the BCC algorithm, we used the Statistical Accounts of Scotland [17]. This collection of census-like data was printed in 1799, with reused metal type on wooden blocks. The age of the paper, combined with the low-quality type and press-work, presents challenges for OCR. The typeface is characterized from modern business documents by the use of the “long s ” (f) and a high occurrence of ligatures **ff**, **ffi**, **ffl**, etc.) Table 2 shows the distribution of the types of characters in the collection. The manually-generated groundtruth data also makes this collection valuable for research.

Table 2: Distribution of character types in the sample data. Note that failure to deal with broken and touching characters gives a maximum possible accuracy of 81.3%.

| | |
|--|-------|
| complete characters | 81.3% |
| broken characters | 10.7% |
| legit. broken characters (i, j, ;, : etc.) | 6.2% |
| touching characters | 1.8% |

The results below were obtained by training the classifier using five pages, and then testing the algorithm on five additional unseen pages with similar typeface.

If the symbol classifier only has knowledge of the complete characters in the image, BCC correctly finds 71% of the broken characters in our test data. By training the symbol classifier with examples of broken characters that were manually identified, BCC correctly finds 91% of the broken characters.

BCC also performs well with legitimately broken characters, such as i, j, ; and :. By training the symbol classifier with examples of each of these characters, BCC was able to find and join 93% of the legitimately broken characters. This renders further procedural programming of heuristic rules (such as to attach i's to dots) unnecessary. Therefore, it is easy to support new character sets that have other legitimately broken characters, such as the Greek majuscule xi (Ξ).

As for touching characters, our approach catches and correctly splits 93% of the touching characters in the dataset. Note, however, that touching characters had a much lower occurrence than broken characters.

3.3 Improving learning classifiers with clustering

Learning classifiers provide a flexible basis for the creation of document recognition tools that adapt to the specific set of characters of symbols present in a document [18]. Unfortunately, the manual labeling of thousands of examples that is required to train a learning classifier is time-consuming and error-prone. In the case of cultural heritage materials, which may be degraded or in obscure languages, training may require specialized knowledge, making it more difficult to find users capable of training a system [1]. Additionally, verifying the training data after labeling is important to ensure the accurate performance of the classifier. Finally, the automatic evaluation of the performance of a learning classifier is required for large digitization projects. This section presents the use of clustering for the pre-labeling of symbols to shorten training time and for

the verification of the data after training.

3.3.1 Clustering Overview

Clustering is the organization of a collection of input patterns into clusters based on similarity. In the context of document recognition, a clustering algorithm would be presented with the symbols from a document and would, ideally, group all of the same symbols together. Clustering is a large field with many different goals and techniques (see [19] for a review of clustering techniques and applications). In this application of clustering to multi-lingual document recognition, graph-theoretic clustering was chosen because it does not require the target number of clusters to be known, is deterministic, and has reasonable algorithmic complexity for the size data considered [20, 19].

3.3.2 Graph-Theoretic Clustering

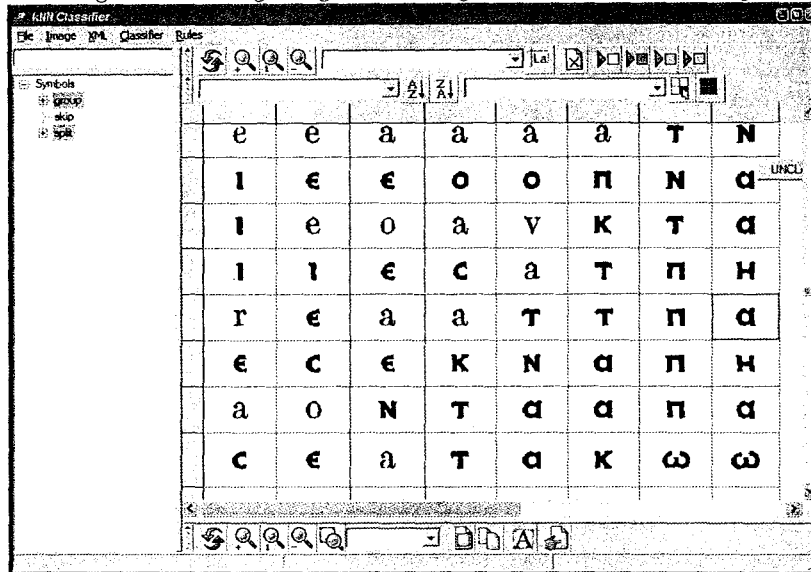
Graph-theoretic clustering operates by creating a minimum spanning tree (MST) of a set of points and deleting edges in the graph based on some criterion [20]. The sub-graphs created by the deletion of the edges represent the clusters. In this application, the point set is created by segmenting an image into symbols and measuring the distance between all of the unique pairs of symbols. The distance measurement, which operates on a set of features representing the symbol, used in these experiments is either simple Euclidean or Manhattan. The criterion used for the deletion of edges is the standard deviation of the current edge weight to the mean of the edge weights of the edges within a local window. Experimentation reveals that using a standard deviation of above 1.5 and window size of 2 to 4 nodes from the current edge yields good results on a variety of documents.

3.3.3 Pre-classification clustering

Instance-based learning classifiers, like the nearest-neighbor classifier, leverage the knowledge of domain-experts by allowing them to train the classifier by identifying (labeling), a set of symbols that the classifier uses to identify subsequent symbols. This allows the classifier to be used to identify almost any symbols without additional software development. Unfortunately, the time required to train learning classifiers can be a major drawback.

At the beginning of the training process the system, by definition, has no knowledge of the symbols that are about to be labeled. It is not possible, therefore, to present the symbols in a way that makes their relationships apparent to the user, making the task of training take significantly longer than if the symbols were already grouped. Figure 8 shows the initial state of the Gamera training tool.

Figure 8: The beginning of a training session without clustering.



Using clustering to group the symbols together before presenting them to the user can reduce the training time, however. This reduces the problem of training to labeling groups of symbols and has the potential to reducing training mistakes. Figure 9 shows the initial state of the Gamera training tool when pre-labeling clustering is used.

3.3.4 Post-Training Verification

The effectiveness of a learning classifier is dependent on the quality of the training set. Clustering can be used to help a user verify the manual labeling of symbols. Each group of manually trained symbols can be clustered and the results presented to the user. Because any incorrectly labeled symbols will appear as a separate cluster in the results it is much easier for the user to identify mistakes than if presented with all of the symbols from a group without clustering. It is important to note that this verification process cannot be fully automatic, even if the clustering algorithm performed without errors, because a user may choose to label visually dissimilar glyphs as the same symbol. For example, the same letter in multiple typefaces might be labeled as one symbol. This flexibility in the training is an important aspect of many learning algorithms and should be preserved.

4 Conclusion

The Gamera system is well on its way to being a complete framework for the development of recognizers for a broad range of historical documents. Our own use of the system has helped us find weaknesses

in the framework that we've subsequently improved. We are also pleased by how much more efficient and less frustrating programming in the Gamera framework is relative to building standalone applications from scratch. Our success in recent applications with real-world data is a testament to the utility of our approach.

Acknowledgments

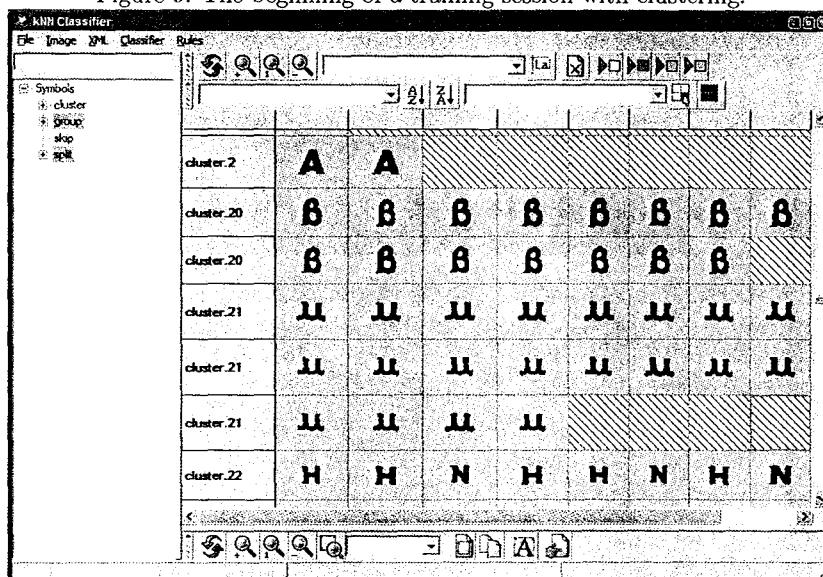
The second phase of the Levy Project is funded through the National Science Foundation's Digital Library Initiative 2 (Award #9817430), an Institute of Museum and Library Services National Leadership Grant, and support from the Levy family.

We also wish to thank the rest of the Digital Knowledge Center staff, (Teal Anderson, G. Sayeed Choudhury, Tim DiLauro, Mark Patton and Cynthia York), for their assistance with this paper.

References

- [1] Droettboom, M., K. MacMillan, I. Fujinaga, G. S. Choudhury, T. DiLauro, M. Patton, and T. Anderson. 2002. Using the Gamera framework for the recognition of cultural heritage materials. *Joint Conference on Digital Libraries*. 11-7.
- [2] Rossum, G. 2002. Python language reference. F. L. Drake, Jr., ed. <http://www.python.org>
- [3] Berezhny, L., J. Elkner, and J. Straw. 2001. Using Python in a high school computer science

Figure 9: The beginning of a training session with clustering.



- program: Year 2. *International Python Conference*. 217–23.
- [4] Cover, T. and P. Hart. 1967. Nearest neighbour pattern classification. *IEEE Transactions on Information Theory*. 13(1): 21–7.
- [5] Holland, J. H. 1975. *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor.
- [6] Schemenauer, N. 2002. *A simple neural network library* (bpan.py). Computer software. <http://arctrix.com/nas/>
- [7] Hoos, H. H. and K. Hamel. 1997. GUIDO music notation version 1.0: Specification part I, Basic GUIDO. Technical Report 20, Technische Universität Darmstadt.
- [8] Dunn., R. *wxPython toolkit*. Computer software. <http://www.wxpython.org>
- [9] Jähne, B., H. Haußecker, and P. Geißler. 1999. Reusable software in Computer Vision. *Handbook on Computer Vision and Applications*. New York: Academic Press.
- [10] Bøe, S., T. Lønnestad, and O. Milvang. 1998. XITE: X-based image processing tools and environment: User's manual, version 3.4. Technical Report 56, Image Processing Laboratory, Department of Informatics, University of Oslo.
- [11] Puckette, M. 1988. The Patcher. *International Computer Music Conference*. 420–9.
- [12] Clocksin, W. F. and C. S. Mellish. 1981. *Programming in PROLOG*. Berlin: Springer.
- [13] Trier, Ø. D. and A. K. Jain. 1995. Goal-directed evaluation of binarization methods. *IEEE Transactions on Pattern Analysis & Machine Intelligence*. 17(12): 1191–201.
- [14] Kass, M., A. Witkin, and D. Terzopolous. 1987. Snakes: Active contour models. *International Conference on Computer Vision*. 259–68.
- [15] Harding, S. M., W. B. Croft, and C. Weir. 1997. Probabilistic retrieval of OCR degraded text using *N*-grams. *European Conference on Digital Libraries*. 345–59.
- [16] Fujinaga, I., B. Alphonse, B. Pennycook, and K. Hogan. 1991. Optical music recognition: Progress report. *International Computer Music Conference*. 66–73.
- [17] *Statistical Accounts of Scotland*. 1799. <http://edina.ac.uk/statacc/>
- [18] MacMillan, K., M. Droettboom, and I. Fujinaga. 2001. Gamera: A Python-based toolkit for structured document recognition. *Tenth International Python Conference*.
- [19] Jain, A. K., M. N. Murty, and P. J. Flynn. 1999. Data Clustering: A Review. *ACM Computing Surveys*. 31(3): 264–323.
- [20] Zahn, C. T. 1971. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*. 20: 68–86.

3D Methods to aid Handwriting Analysis and OCR

**Dr. Anshuman Razdan¹ (ar@asu.edu),
John Femiani (john.femiani@asu.edu) and
Dr. Jeremy Rowe (jeremy.rowe@asu.edu)**

Abstract

The well known problems in OCR and handwriting analysis that confound efforts to automate translation include foreground colors mixed with a background pattern such as numbers on bank notes, lack of separation of letters, varying letterforms, rubrics and abbreviations, and, accents and omitted characters for handwritten documents. Within each letter such as “*l*” there is the problem of deciphering the movement of pen and recognizing defining characteristics of the loop. Current methods are limited to two dimensional (2D) interrogations such as feature vectors of the bitmap of the digitally scanned documents.

We have developed a novel method whereby we convert the 2D image data into a three-dimensional (3D) volumetric data. Filtering and subsequent surface extraction results in 3D surfaces that correspond to the characters on the paper and convey additional information that can assist in identifying characters. 3D geometric interrogation of these surfaces such as *crest lines* and medial axis can reveal very interesting properties. Characters such as “*l*” clearly denote the overlapping loop structure which can be extracted easily and the whole curve can be parametrized accurately as a chain code. The research, though still in early development, has produced some encouraging results which will be shared at the conference. In case of bank note problem, this technique can automatically extract individual numbers from the background pattern. The method has potential for analysis and characterization of individual handwriting based on the curvature analysis of the 3D surfaces. These features have the potential to be invariant to rotation and placement on the paper i.e. separating character (sequences) from different rows intermingled (written over) with those from another row. This method opens up the field of analysis and OCR from a 3D perspective and highly developed mathematics for 3D geometric *feature recognition* can now be applied to these problems.

¹ Corresponding author. MC 5906 PRISM, Arizona State University, Tempe, AZ 85287-5906, (480) 965-0483 (razdan@asu.edu)

Automated Reading of Free-Form Handwriting in Images, The Past and One Proposed Future

Joanna Fancy
Higherglyphics, 726 Idaho Ave, Santa Monica, CA 90403
j.fancy@ieee.org

Abstract

Developing software to read natural handwriting is difficult, even for pen-entry data. Reading handwriting in images is far more difficult. Except for very narrow applications, we have not solved this problem, and the need is pressing. In the intelligence arena, for instance, intercepted messages may go unread until it is too late. And mountainous archives of paper documents grow daily bigger – medical records, army field notes, historical manuscripts – unreadable and unsearchable by automated means.

In attacking this problem, image-based reading, various combinations of statistical brute force and heuristic finesse have been tried. The brute force is generally applied by a trained neural network, which can compare a character or word to examples in a vast database. Finesse comes into play because writing samples must be described before comparisons can be made. Unfortunately, effective features are not obvious and those that have been tried have not been sufficiently powerful or robust. As a result, while current recognition systems work well on single, isolated characters, they can be applied to whole words only if context drastically limits the universe of possible solutions. For this reason, recognizers fail under most real world scenarios, and progress has been so slow that most researchers have moved on to other problems.

To achieve a real success, it will be necessary to solve the problem of sequence. We must reconstruct from a static image the pen's original path, point by point, from pen down to pen up. Sequence is the raw data captured by a pen and tablet device, and it accounts for the success of pen-based, as opposed to image-based, recognition. Even for a human reader, following sequence, the movement in writing, may be at least as important as seeing the image the pen has produced; sequence contains information the image does not. An interesting demonstration of this principle was recorded by Aleksander Luria, the great Russian neuropsychologist. Luria found that a patient whose wartime brain injury had deprived him of the ability to read could retrain himself simply by tracing over with a finger the words he was trying to make out. His kinesthetic memory, uninjured, allowed him to decipher the pen's path and recognize the letters traced, even though they still made no sense to him as images.

Some steps have been taken toward solving the sequencing problem. For instance, a writing stroke can be broken into manageable pieces and each tangled knot treated as a graph to be solved with a variety of the traveling salesman algorithm. The heuristic argument for this approach is that writing tends to be economical and the shortest route that fully traverses a path is likely to be the one actually taken. Valid as this notion may be, the graph-theoretic solution does not yield true sequence. It cannot reveal point by point direction, and even in ordering sections of the stroke some decisions are arbitrary.

It is the contention of Higherglyphics that not enough has been done to exploit heuristic finesse in solving either the crucial sequencing or the overall reading problem. The best rules of thumb have always been based on the way handwriting is produced. The simplest example is that a stroke will generally begin at the left and proceed rightward. However, all rules have exceptions, and those few which have been tried have been applied too broadly. There are many subtle and untried heuristic principles which must be added to the mix before training-based brute force can be effective.

Higherglyphics has devised a system, using scanned, unconstrained handwritings, which takes a novel approach to the reading problem. The promise of the Higherglyphics system has been confirmed repeatedly by those who have evaluated it. At the request of NSA, it was reviewed and recommended by IDA/CCR and was to have been funded by the NSA IDEA Program, until IDEA was itself defunded. It was then "deemed selectable" by ARDA's AQUAINT Program, though it could not be funded because it did not fit AQUAINT's mission. Most recently, it was examined by TSWG; 12,700 ideas were submitted in response to the TSWG counter-terrorism BAA, 627 White Papers were requested, 50 projects were immediately funded and 150 were put on a waiting list, the Higherglyphics proposal among them.

The next step is to program a prototype. Whether or not Government funding can be found for this project, Higherglyphics is intent on implementing and fully testing its system. We will report results as they become available.

THE VIDEO SPECTRAL COMPARATOR 2000HR

*Gregg Mokrzycki,
Forensic Document Examiner
Questioned Documents Unit
FBI Laboratory
2501 Investigation Parkway
Quantico, VA 22135*

ABSTRACT

The Video Spectral Comparator 2000HR (VSC) is an imaging device that allows a document examiner to analyze inks, visualize hidden security features, reveal alterations on a document, and render visible information that is invisible to the naked eye. With the combined power of a desktop computer, a high resolution digital camera, various light sources, and a full array of excitation/barrier filters, the VSC enables an examiner to easily switch between various lighting and filter combinations to extract information from evidence.

FBI document examiners have used the VSC on numerous cases ranging from the recent Sniper and 9/11 attacks to the Anthrax investigation. Since the VSC is able to reveal information that is not visible to the naked eye, it is especially useful in uncovering details that may have been overlooked by a criminal and therefore makes it an indispensable addition to a document examiner's arsenal of forensic tools. This presentation will show the function of the machine and how it is used working actual cases.

Keywords: VSC2000HR, Video Spectral Comparator, alternate light source, questioned documents

Learning Objective and Outcome:

To reveal how the Video Spectral Comparator 2000HR is utilized by the FBI Questioned Documents Unit.

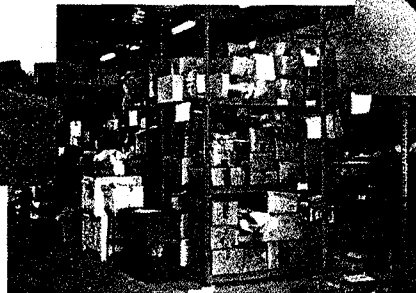
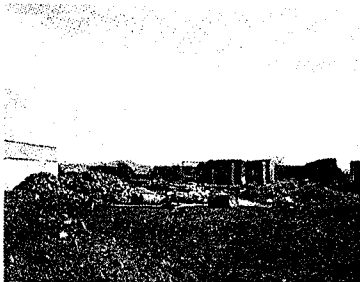
Document Layout Analysis

Thomas M. Breuel, PARC

In this demonstration, I will show the application of research results and algorithms given in the companion paper to a number of real-world documents. This includes tools for document image normalization and cleanup, high-accuracy page de-skewing, text line finding, white space analysis and column finding, reading order determination, and overall document layout analysis.

Document Exploitation Functions

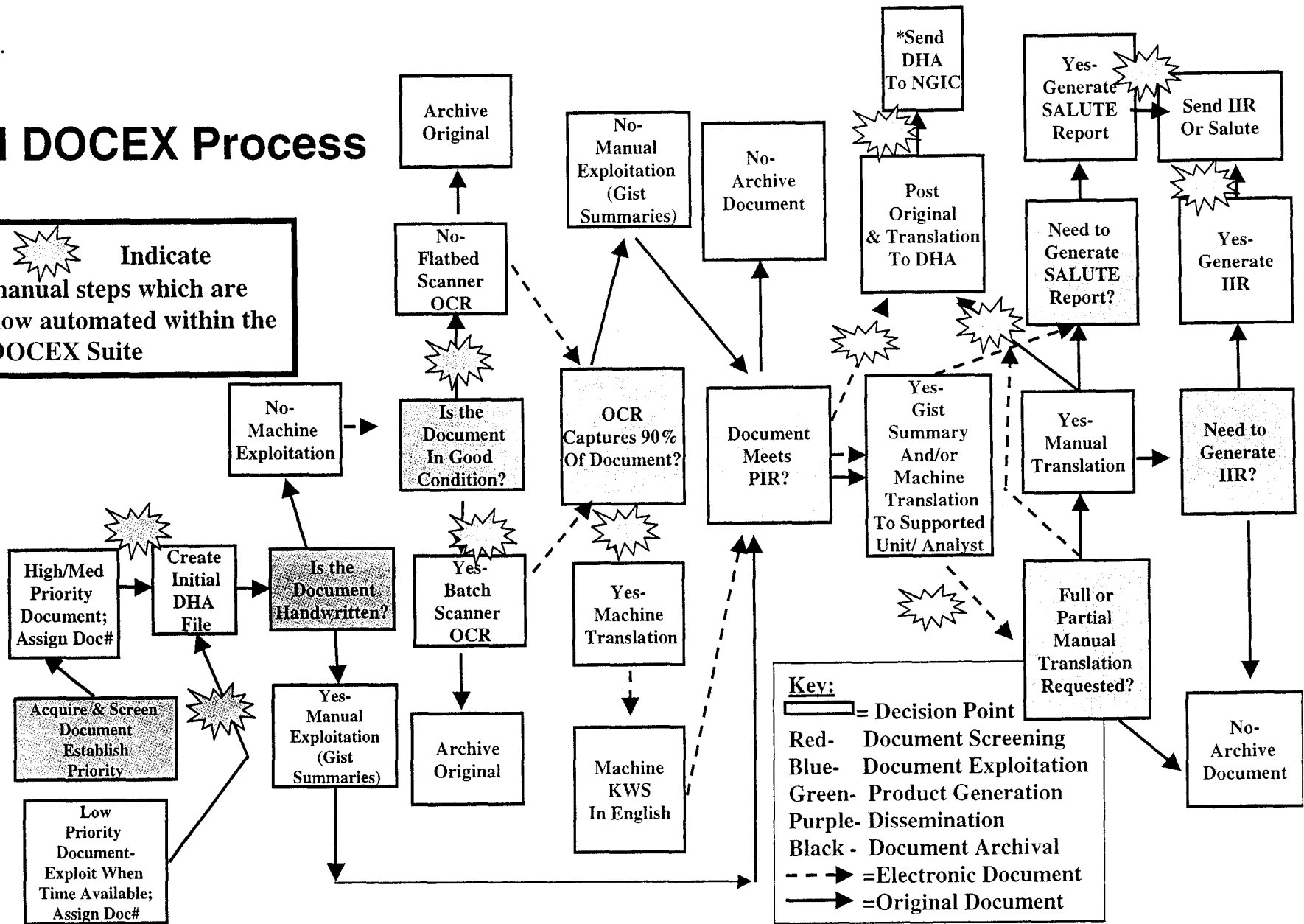
- Collection (EPW pocket litter, paper documents, laptops, ...)
- Digital Capture (Photo, Scan, Import)
- Translation
- Screening
- Content Categorization
- Analysis
- Reporting



MI DOCEX Process

Indicate manual steps which are now automated within the DOCEX Suite

297



DOCEX Suite Objectives – “What”

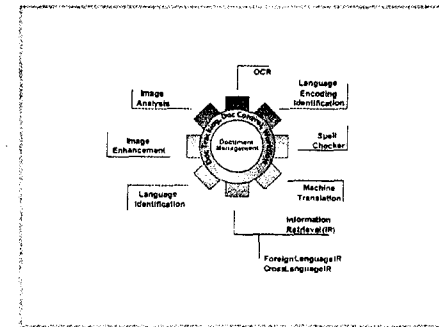
- **End-to-End Suite for the Document Exploitation Life-Cycle**
 - Support Paper and Electronic Documents & Multimedia
 - Encode ‘Expert’ DOCEX Processing Knowledge via Defined Workflows (to automatically support the ***general, logical, and systematic exploitation process*** resulting from our DOCEX experiences/lessons learned)
 - Leverage Current State of OCR/MT via Machine Assisted Identification, Translation, Screening & Prioritization (***find out what we have***)
 - Provide a Managed and Shared Repository with Collaboration Tools

DOCEX Suite Objectives – “What”

- **Integrate with All Levels and Components (Mud to Space)**
 - Support Local, Remote and Theater Access
 - Automation Assistance for Human Exploitation
 - Respond to Changing Priorities as PIRs Change (Time/Echelon)
 - Respond to Dynamically Changing Collections
 - Automatic Notification and Delivery of Relevant Documents to Analysts, Report Writers, Translators, etc.

DOCEX Suite Objectives – “How”

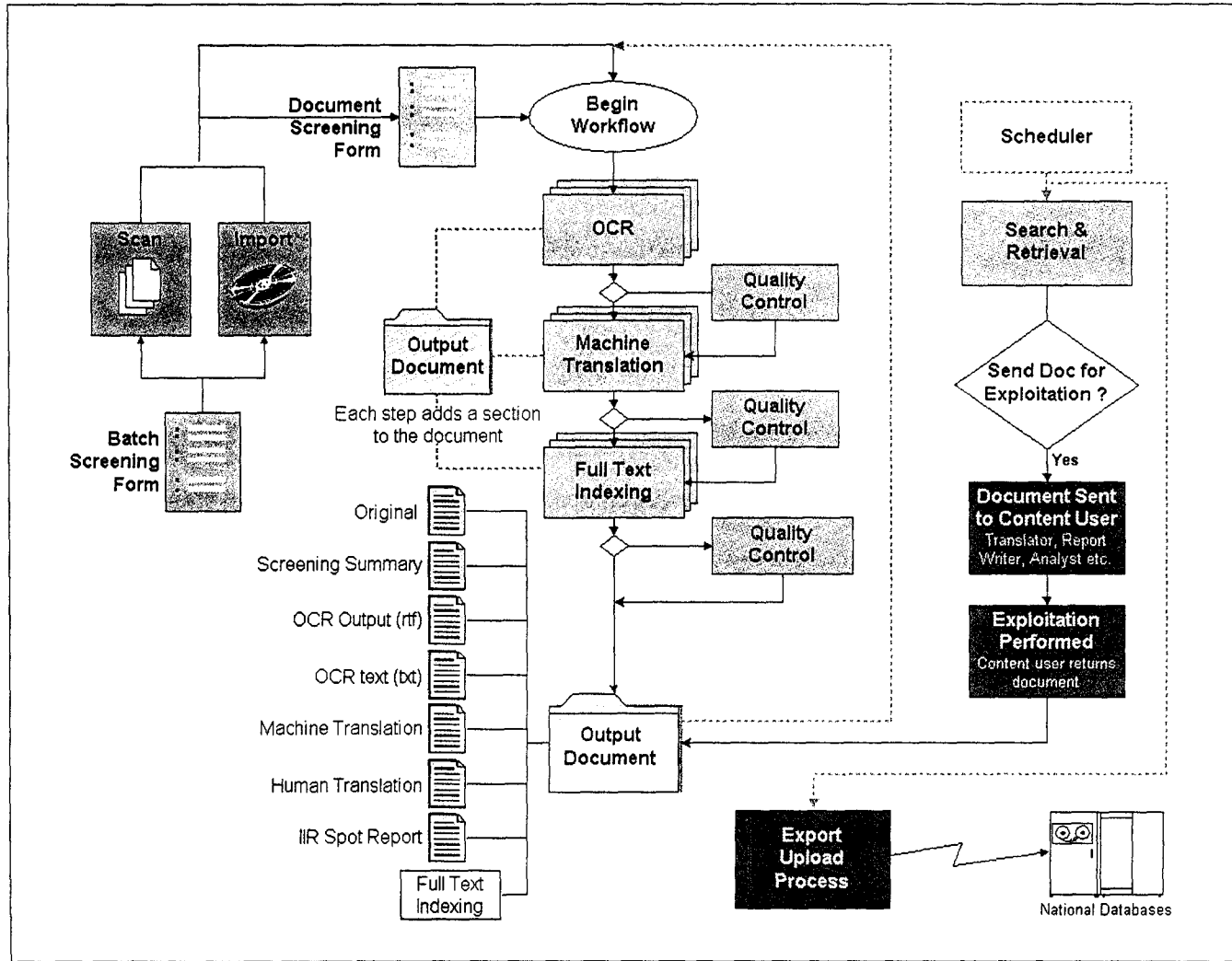
- **Leverage DoD DOCEX Lessons Learned**
- **Leverage FALCON & Associated Components**
- **Leverage LASER Technology**
 - Image Analysis & Enhancement
 - Language & Encoding Identification
 - OCR
 - Machine Translation
 - Information Retrieval (Foreign Language and Cross-Lingual)
- **Leverage Other Mature and Emerging Products**
 - HighView COTS, GOTS and Advanced DoD & IC Development
 - Other DoD & IC Technology
 - Other 3rd Party Industry Standard Products
- **Continual augmentation of DOCEX as technology improves**



DOCEX Suite – Current Capabilities

- **Document Capture and Screening** (batch and document level)
 - Includes automatic docking and synchronization
- **Workflow** (Capture all the way through Exploitation)
- **Language Support**
 - OCR – Scansoft, Sakhr, Abbyy (187 Languages – no CJK or Thai)
 - MT – Transphere, Gister, Systran (47 Languages)
 - Encoding Identification and Transformation -- Hotspot, Transcoder
- **Search, Retrieval, and Notification** (via e-mail & defined workflows)
 - Full Text Indexing (Latin Script) and Categorization (English)
 - Retrieval via Screening Data, Document Content, Translation Status, etc.
 - Packaged Query Execution
- **Document Management** (Check-In/Out, Versioning, Audit, etc.)
- **Remote Web Access** (for Translators, Analysts and Others)
- **Upload to Other Systems & Levels** (National databases)

DOCEX Workflow



DOCEX Suite – Future Enhancements

- **Extend Automation for Intelligence Information Discovery**
 - Scheduler for Automatic Notification of Documents of Interest
 - Defining User/Group Interest Profiles (PIRs Time/Echelon)
 - Advanced Hot/Word & Phrase Management (NSA)
 - Utilization of Military Terms/Thesauri
- **Finer Grained Control of MT/OCR** (Control Functions & MT Dictionaries)
- **Custom Exploitation and Management Reports**
 - Spot Report (SALUTE) Template
 - Summarizations by Screening Categories by Date, Location, etc.
- **Mapping of Document Locations** (via Geocode and Metadata)
- **Forensic Integration** (safe integration with 'dirty' electronic media)
- **Non-Latin Script Full Text Indexing & Retrieval** (both Text & Themes)

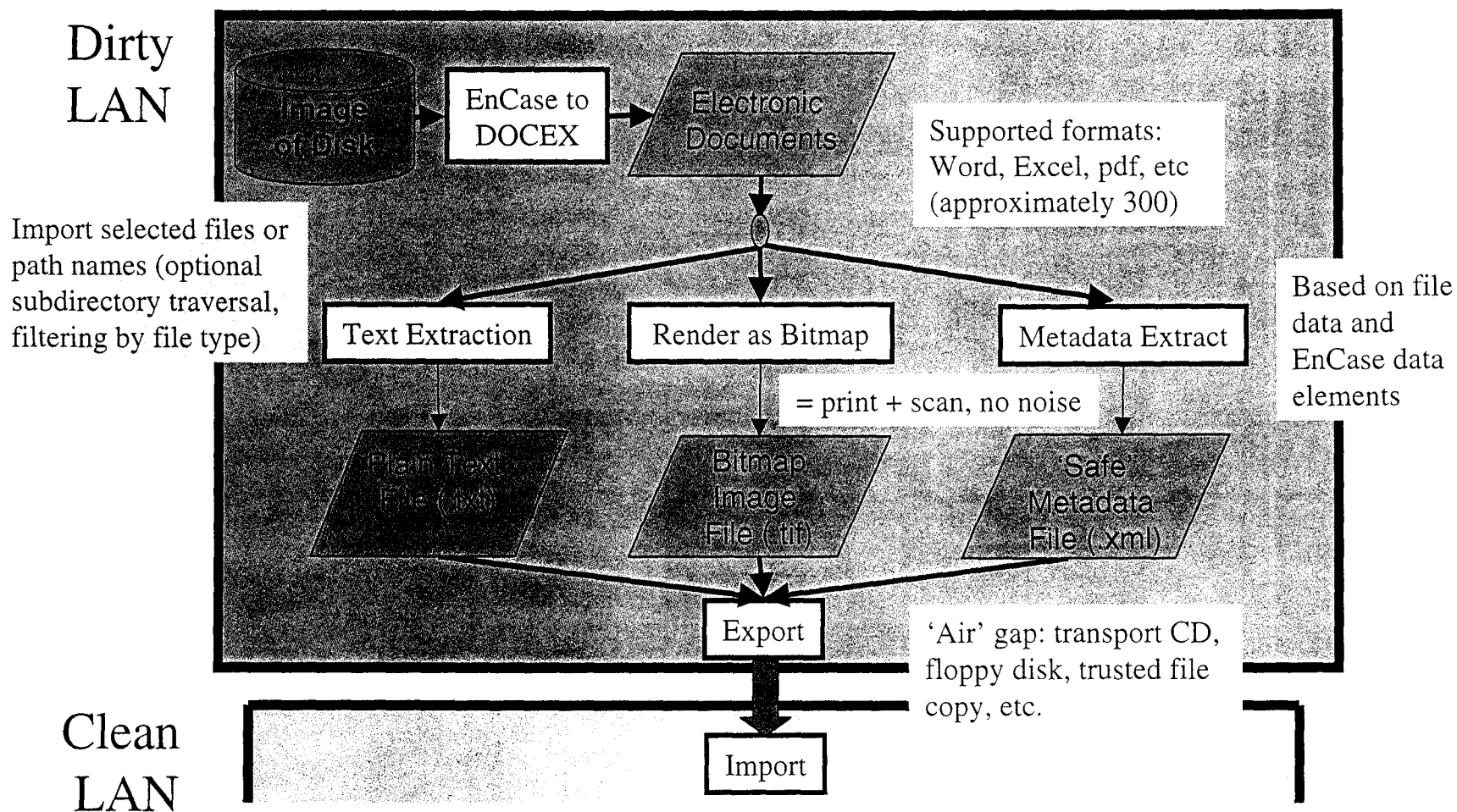
DOCEX Suite – Future Enhancements

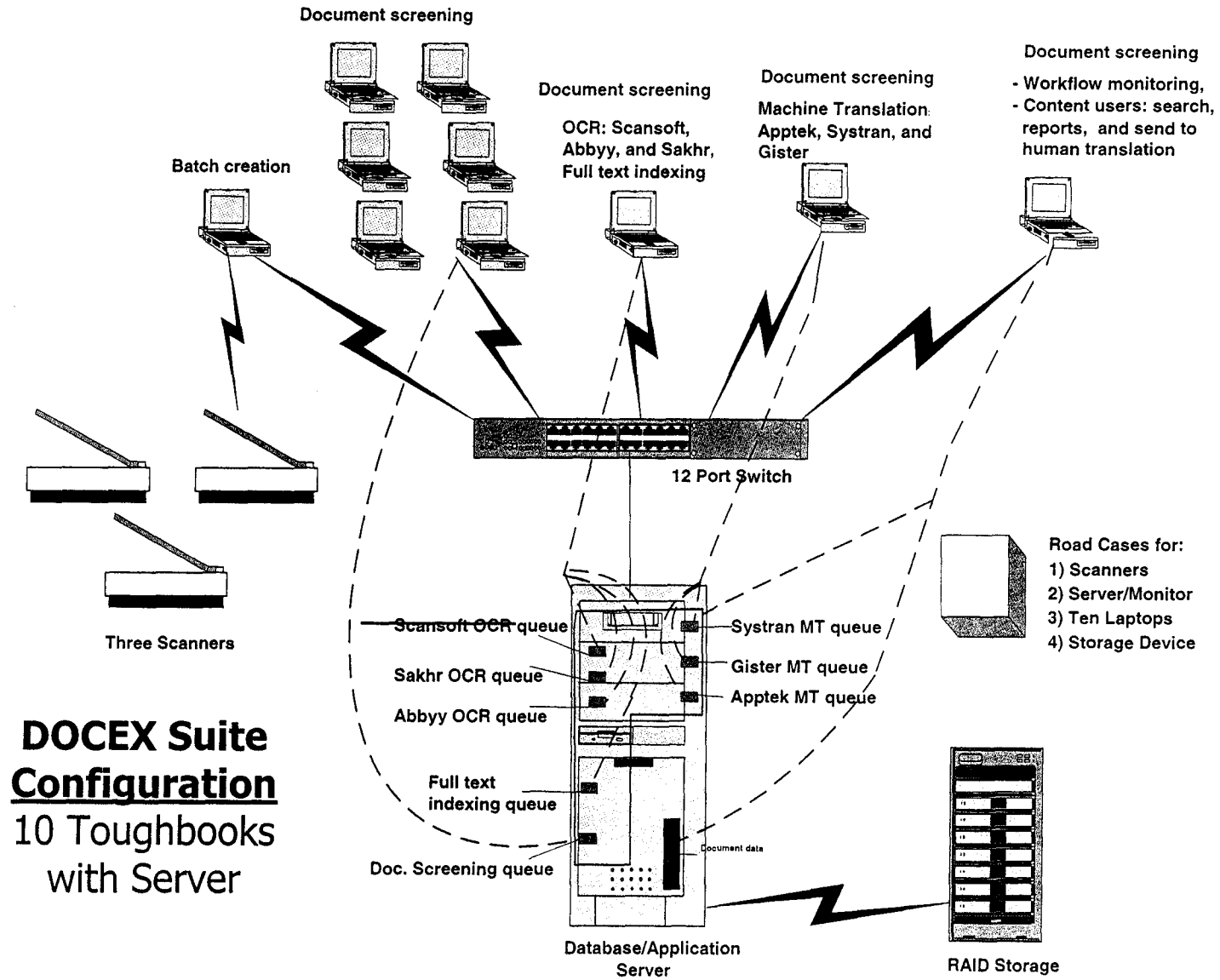
(continued)

- **Continued LASER Component Integration**
 - Additional OCR & MT Engines + Memory Translation & Hybrids
 - Image Analysis & Enhancement
 - Foreign & Cross Lingual Information Retrieval
 - Topic/Category Visualization (known and unknown)
- **Biometric Integration and Evidence Protection** (fingerprint, iris, etc.)
 - Integrated with Documents (pocket litter, etc.)
 - Watermarking of Images
- **Named Entity Extraction & Link Analysis**
- **Video & Audio Transcription, Indexing, and Retrieval**
- **Analysis/Review/Training of MT & Linguist Performance**
 - Community Assessments (Document Level and Aggregations)
 - Audit Trail (Workflow, Document, Metadata)

DOCEX Suite - Planned (Forensic Integration)

305





DOCEX Suite Configuration 10 Toughbooks with Server

SCANSOFT ASIAN LANGUAGE OCR CAPABILITY

*Tom D'Errico
Manager of Government Operations
ScanSoft, Inc.
9 Centennial Drive
Peabody, MA 01960*

About ScanSoft: ScanSoft, Inc. is the leading supplier of imaging, speech and language solutions that are used to automate a wide range of manual processes - saving time, increasing worker productivity and improving customer service.

ScanSoft's imaging products include OmniPage (OCR), PaperPort (document management), OmniForm (electronic forms). ScanSoft's speech & language solutions include Dragon Naturally Speaking (speech recognition).

Imaging toolkit - ScanSoft's Capture Development System is a complete solution for adding OCR, ICR, Barcode, OMR and PDF capabilities to applications. Capture Development Systems recognizes 114 Latin and Cyrillic languages in addition to Japanese, Chinese and Korean.

Demonstration - ScanSoft will demonstrate its Japanese, Chinese and Korean OCR technologies.

Groundtruth Image Generation from Electronic Text (Demonstration)

David Doermann and Gang Zi
Laboratory for Language and Media Processing,
University of Maryland, College Park MD 21043, USA
{doermann, gzi}@umiacs.umd.edu

Abstract

The problem of generating synthetic data for the training and evaluating of document analysis systems has been widely addressed in recent years. With the increased interest in processing multilingual sources, there is a tremendous need to be able to rapidly generate data in new languages and scripts, without the need to develop specialized systems. We have developed an approach that uses language support of the MSWindows operating system combined with custom print drivers to render tiff images simultaneously with windows Enhanced Metafile directives. The Metafile information is parsed to generated zone, line, word, and character groundtruth including location, font information and content in any language supported by Windows. The processing is embedded in a collection of tools for data generation, groundtruthing, degradation and evaluation. The discussion here focuses on the Groundtruth Generator.

1 Introduction

Generating synthetic document images and symbolic groundtruth files in large scale has become a recent focal point for training OCR algorithms and evaluating the performance of OCR systems [1], [2], [6]. Typically, training and evaluation require the groundtruth data to be keyed in manually from the scanned image, but this is often a prohibitively labor-intensive and error prone process. Furthermore, it may require domain experts, especially for processing multi-lingual documents.

In this text we present a methodology for generating synthetic document images and symbolic groundtruth files automatically by using a custom print driver and metafile information. We give a brief survey of related work, describe the system architecture, and present the main component of our system: the groundtruth generator.

2 Related work

Using synthetic data has many advantages including the rapid generation of datasets at low cost, easy control of degradations models and parameters, and convenient testing of the same underlying documents with different corruption methods [1]. To generate synthetic data, many methods have been proposed. In [3], the authors presented an approach to get the noise-free document images from DVI (device independent format) files. However, the requirement of DVI files and LATEX typesetting

limits the practical application in many cases. In [4], the authors present an approach to propagating groundtruth information from an original collection allowing the reuse of groundtruth information and [5,6] extending previous work on the use of degradation models for data generation.

Choosing suitable corpora for the evaluation plays a crucial part in evaluating an OCR system. A representative corpus should have all characteristics of the target applications. Although a number of datasets have been created, they are typically not appropriate for all applications, but nevertheless, allow focused evaluation. For example, English technical journals are used in the UW dataset and magazines; Spanish newspapers, and English and German business letters are used in the UNLV OCR evaluation set. Our approach allows users to supplement traditional groundtruth with images and groundtruth generated from electronic text, formatted in a way that is representative of the domain.

3 System Architecture

The system architecture is shown in Figure 1.

Starting from the structured electronic files, such as MSWord or HTML files, we import the source to a renderer and generate the noise-free images and the groundtruth files. The system uses MSWindows print drivers, so the document content can be rendered the same way to many different devices. The degraded images can be obtained from the ideal images through a degradation model, or by physically degrading (printing, scanning, faxing, etc) a hard copy. Finally, the synthetic images and groundtruth files can be used for training and evaluation. One application of our work is to study the downstream effect of OCR degradation of information retrieval (IR), and machine translation (MT) systems.

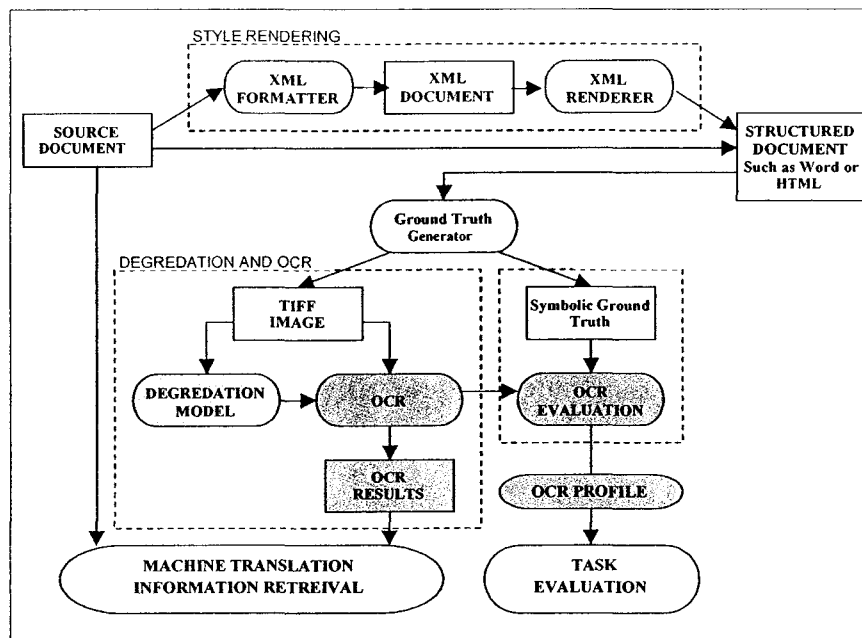


Figure 1: OCR evaluation system architecture

4 Groundtruth Generator

In our system, a groundtruth generator (GTG) is used to obtain the synthetic noise free images and the symbolic groundtruth files. First, the structured documents, such as HTML or MSWord files, are input to the GTG system. Image files and the metafiles are generated via a custom printer driver. By parsing the resulting metafiles, we obtain symbolic and layout information, and generate groundtruth in various formats. The synthetic images and layout information is used to create overlaid images, where the bounding boxes are displayed at the character, word, line, and zone level.

Three kinds of groundtruth files are generated in GTG: *Standard*, *Raw* and *Structured*.

- **Standard groundtruth** contains basic information about the size of the reference pages, fonts used in the document, the character set of the content, and zone, line, word and character information where appropriate. For each zone, we identify the type of zone (Text, Image or Graphic), for each word, we identify the font, and for each character, we identify the font glyph and Unicode text.

```
CONTENT (pixels): (629,264,1863,2279)
PAGE SIZE (mm): (0,0,213,273)
RESOLUTION: 301 dpi
Font 0: Times New Roman, ARABIC_CHARSET
Font 1: Times New Roman, ANSI_CHARSET
Font 2: Times New Roman, ANSI_CHARSET
Font 3: Bold Times New Roman, ARABIC_CHARSET
Font 4: Bold Times New Roman, ARABIC_CHARSET
Font 5: Bold Courier New, ARABIC_CHARSET
ZONE 0: (1248, 2237, 1267, 2279) T
LINE 0: (1248, 2237, 1267, 2279)
WORD 0: (1248, 2237, 1267, 2279) 2
CHAR 0: (1248, 2237, 1267, 2279), 50 , 00 32, 50
ZONE 1: ( 629, 336, 1863, 1205) T
LINE 0: (1446, 336, 1863, 406)
WORD 0: (1446, 340, 1554, 406) 5
CHAR 0: (1446, 340, 1482, 406), 1575 , 06 27, 909
CHAR 1: (1482, 340, 1518, 406), 65194 ,fe aa, 938
CHAR 2: (1518, 340, 1554, 406), 65191 ,fe a7, 935
CHAR 3: (1554, 340, 1589, 406), 32 ,00 20, 3
WORD 1: (1589, 340, 1732, 406) 5
```

- **Raw groundtruth** files are in Unicode format or in original coding format. These files contain only the character content and can be used for OCR evaluation. The encoding will be identical to the original encoding used to generate the structured document.
- **Structured groundtruth** files include HTML and XML files. The HTML files can be used to check whether the groundtruth file is the same with the original document by simply viewing the results in a browser. The XML files are used for data exchange or storage.

Because the groundtruth files are parsed from metafiles, as long as True Type Font (TTF) files are used, data from any character set can be created. We have tested our system on dozens of languages, including Arabic, Chinese, Farsi, Japanese, Thai, Hindi and Korean. From this point of view, our system provides a universal framework to generate groundtruth files for multi-lingual documents.

The synthetic images are noise-free images and can be generated at different resolutions. Those images can be degraded on pixel level with a parameterized model, or degraded on page level with noise templates.

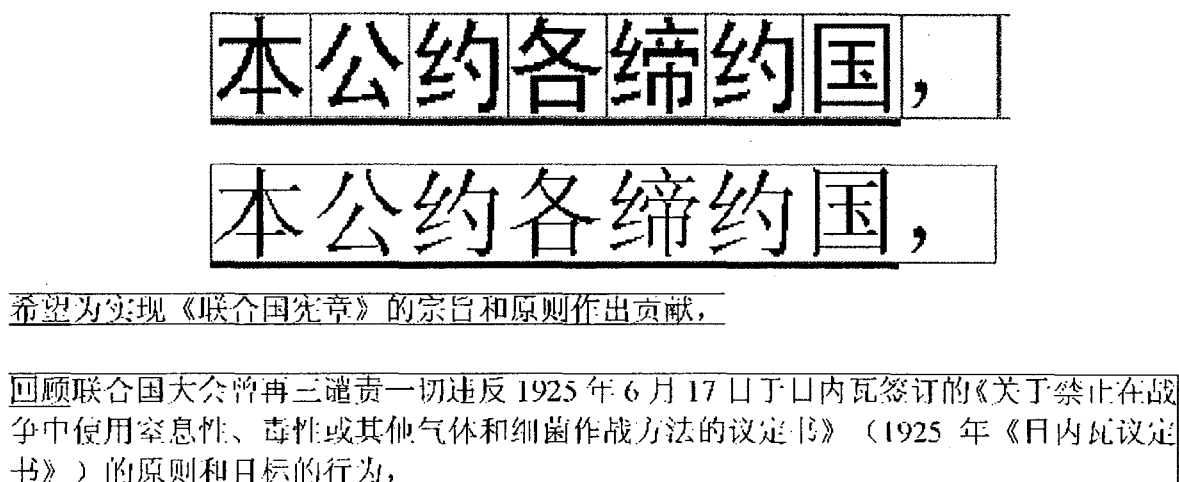


Figure 2: Overlaid images in character, line, and zone level

For more information about the system, please contact the authors.

References

- [1] D. Doermann, and S. Yao. Generating Synthetic Data for Text Analysis Systems. *SDAIR*, pages 449-467, 1995 .
- [2] Tin Kam Ho, Henry S. Baird, Evaluation of OCR Accuracy Using Synthetic Data, *SDAIR95*, 1995
- [3] T. Kanungo, R.M. Haralick, and I.T. Phillips. Nonlinear local and global document degradation models. *IJIST*, 5(4):220-30, 1994.
- [4] T. Kanungo and R.M. Haralick. An automatic closed-loop methodology for generating character groundtruth for scanned documents. *PAMI*, 21(2):179-183, February 1999.
- [5] T. Kanungo, R.M. Haralick, H.S. Baird, W. Stuezle, and D. Madigan. A statistical, nonparametric methodology for document degradation model validation. *PAMI*, 22(11):1209-1223, November 2000.
- [6] T. Kanungo, and P. Resnik. The Bible, Truth, and Multilingual OCR Evaluation. *SPIE Conference on Document Recognition and Retrieval (VI)*, pages 86-96, JAN 1999 .
- [7] D.W. Kim and T. Kanungo. Attributed point matching for automatic groundtruth generation. *IJDAR*, 5(1):47-66, 2002.
- [8] T. Kanungo, etc., "Document Degradation Models: Parameter Estimation and Model Validation", *Proc. of IAPR Workshop on Machine Vision and Applications*, Kawasaki, Japan, 1994, pp. 552-557
- [9] Esko Ukkonen, "Algorithm for Approximate String Matching", *Information and Control* vol. 64, pp. 100-108, 1985
- [10] S.V. Rice, etc., "The fifth annual test of OCR accuracy", *Tech. Rep. TR-96-01*, Information Science Research Institute, University of Nevada, Las Vegas, NV, 1996.

A Generative Probabilistic OCR Model

Okan Kolak^{†,*} Philip Resnik^{†,*}
 Computer Sci.[†], Linguistics[‡], UMIACS^{*}
 University of Maryland
 College Park, MD 20742, USA
 {okan, resnik}@umiacs.umd.edu

William Byrne
 CLSP
 The Johns Hopkins University
 Baltimore, MD 21218, USA
 byrne@jhu.edu

Abstract

In this paper, we present a generative probabilistic optical character recognition (OCR) model that describes an end-to-end process from generation of the true word sequence to the noisy output of an OCR system. The model is designed for use in error correction in a post-processing framework, treating the OCR system as a black-box. The focus of the system is to make OCR output more useful for computer usage. We present a finite-state machine based implementation of the model, and demonstrate its ability to significantly reduce word and character error rates.

1 Introduction

Despite the increase in the amount of text stored in electronic form, vast quantities of information is still available primarily, or only, in print. Some important applications of the NLP technology, such as rapid, rough document translation in the field [1] or information retrieval from scanned documents [2], can depend heavily on the quality of optical character recognition (OCR) output. The alternative of using the raw images remains to be elusive for practical systems [3].

Unfortunately, the output of commercial OCR systems is far from perfect, especially when the language in question is resource-poor [4]. And efforts to acquire new language resources from hardcopy using OCR [5] face something of a chicken-and-egg problem. The problem is compounded by the fact that most OCR system are black boxes that do not allow user tuning or re-training

In this paper, we describe a complete, probabilistic, generative model for OCR, motivated specifically by (a) the need to deal with monolithic OCR systems, (b) the focus on OCR as a component in NLP applications, and (c) the ultimate goal of using OCR to help acquire resources for new languages from printed text. After presenting the model itself, we discuss the model's implementation, training, and its use for post-OCR error correction. We provide evaluation results for error correction; and conclude with a discussion of related research and directions for future work. Kolak et al. [6] provides

a more complete description and evaluation of the work presented here.

2 The Model

We propose a generative model that relate an observable OCR output string O to an underlying true word sequence W . Probability of this relationship, $P(W, O)$, is decomposed by Bayes's Rule into steps modeled by $P(W)$ (the source model) and $P(O|W)$ (comprising sub-steps generating O from W). Each step and sub-step is completely modular, so one can flexibly make use of existing sub-models or devise new ones as necessary. Note that the process of "generating" O from W is a mathematical abstraction, not necessarily related to the operation of any particular OCR system.

We begin with preliminary definitions and notation, illustrated in Figure 1. A true word sequence $W =$

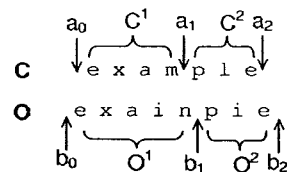


Figure 1: Word and character segmentation

$\langle W_1, \dots, W_r \rangle$ corresponds to a true character sequence $C = \langle C_1, \dots, C_n \rangle$, and the OCR system's output character sequence is given by $O = \langle O_1, \dots, O_m \rangle$.

A segmentation of the true character sequence into p subsequences is represented as $\langle C^1, \dots, C^p \rangle$. Segment boundaries are only allowed between characters. Subsequences are denoted using segmentation positions $a = \langle a_1, \dots, a_{p-1} \rangle$, where $a_i < a_{i+1}$, $a_0 = 0$, and $a_p = n$. The a_i define character subsequences $C^i = \langle C_{a_{i-1}}, \dots, C_{a_i} \rangle$. (The number of segments p need not equal the number of words r and C^i need not be a word in W .)

Correspondingly, a segmentation of the OCR'd character sequence into q subsequences is given by $\langle O^1, \dots, O^q \rangle$. Subsequences are denoted by $b =$

$\langle b_1, \dots, b_{q-1} \rangle$, where $b_j < b_{j+1}$, $b_0 = 0$, and $b_q = m$. The b define character subsequences $O^j = \langle O_{b_{j-1}} \dots O_{b_j} \rangle$.

Alignment chunks are pairs of corresponding truth and OCR subsequences: $\langle O^i, C^i \rangle$, $i = 1, \dots, p$.

Generation of True Word Sequence. The generative process begins with production of the true word sequence W with probability $P(W)$; for example, $W = \langle \text{this, is, an, example, .} \rangle$. Modeling the underlying sequence at the word level facilitates integration with NLP models, which is our ultimate goal. For example, the distribution $P(W)$ can be defined using n -grams, parse structure, or any other language modeling tool.

From Words to Characters. The first step in transforming W to O is generation of a character sequence C , modeled as $P(C|W)$. This step accommodates the character-based nature of OCR systems. It also allows mapping of different character sequences to the same word sequence (case/font variation) or vice versa (e.g. ambiguous word segmentation in Chinese). We output '#' to represent visible word boundaries (i.e. spaces). One possible C for our example W is $C = \text{"This\#is\#an\#example."}$

Segmentation. Subsequences C^i are generated from C by choosing a set of boundary positions, a . This sub-step, modeled by $P(a|C, W)$, is motivated by the fact that most OCR systems first perform image segmentation, and then perform recognition on a word by word basis. Word merge and split errors are modeled at this step as well. A possible segmentation for our example is $a = \langle 8, 11, 13 \rangle$, i.e. $C^1 = \text{"This\#is\#"}$, $C^2 = \text{"\#an\#"}$, $C^3 = \text{"\#ex"}$, $C^4 = \text{"ample."}$ Notice the merge error in segment 1 and the split error involving segments 3 and 4.

Character Sequence Transformation. Our characterization of the final step, transformation into an observed character sequence, is motivated by the need to model OCR systems' character-level recognition errors. We model each subsequence C^i as being transformed into an OCR subsequence O^i , so

$$P(O, b|a, C, W) = P(\langle O^1, \dots, O^p \rangle | a, C, W). \quad (1)$$

and we assume each C^i is transformed independently, allowing

$$P(\langle O^1, \dots, O^p \rangle | a, C, W) \approx \prod_{i=1}^p P(O^i | C^i). \quad (2)$$

Any character-level string error model can be used to define $P(O^i | C^i)$; this is also a logical place to make use of confidence values if provided by the OCR system. For our example C^i , a possible result for this step is: $O^1 = \text{"Tlmsis"}$, $O^2 = \text{"an"}$, $O^3 = \text{"cx"}$, $O^4 = \text{"ample."}$; $b = \langle 7, 10, 13 \rangle$. The final generated string would therefore be $O = \text{"Tlmsis\#an\#cx\#am1e."}$.

Assuming independence of the individual steps, the complete model estimates joint probability

$$P(O, b, a, C, W) = P(O, b|a, C, W)P(a|C, W)P(C|W)P(W) \quad (3)$$

$P(O, W)$ can be computed by summing over all possible b, a, C that can transform W to O :

$$P(O, W) = \sum_{b, a, C} P(O, b, a, C, W). \quad (4)$$

3 Implementation

We have implemented the generative model using a weighted finite state model (FSM) framework, which provides a strong theoretical foundation, ease of integration for different components, and reduced implementation time thanks to available toolkits such as the AT&T FSM Toolkit [7]. Each step is represented and trained as a separate FSM, and the resulting FSMs are then composed together to create a single FSM that encodes the whole model. Details of parameter estimation and decoding follow.

3.1 Parameter Estimation

The parameter estimation methods we devised assume that a training corpus is available, containing $\langle O, C, W \rangle$ triples. Specific parameter estimation methods for each individual step is described in the following sections.

Generation of True Word Sequence. We use an n -gram language model generated using CMU-Cambridge Toolkit.[8] as the source model for the original word sequence. The model is trained on the W from the training data and encoded as a simple FSM. We made a closed vocabulary assumption to evaluate the effectiveness of our model when all correct words are in its lexicon. Therefore, although the language model is trained on only the training data, the words in the test set are included in the final language model FSM.

From Words to Characters. We generate three different character sequence variants for each word: upper case, lower case, and leading case (e.g. $\text{this} \Rightarrow \{\text{THIS, this, This}\}$). For each word, the distribution over case variations is learned from the $\langle W, C \rangle$ pairs in the training corpus. For words with very low or zero occurrence counts, we back off to word-independent case variant probabilities.

Segmentation. Our current implementation makes an independent decision for each character pair whether to insert a boundary between them. The number of boundary insertions are limited to one per word for practical reasons. The probability of inserting a segment boundary between two characters is conditioned on the character pair, and estimated from the training corpus.

Table 1: Post-correction WER and CER and their reduction rates under various conditions

| Conditions | | | | Results | | | |
|---------------------|-----------|-------|-----------|---------|----------|---------|----------|
| LM | WC | SG | EM | WER (%) | Red. (%) | CER (%) | Red. (%) |
| Original OCR Output | | | | 18.31 | - | 5.01 | - |
| Unigram | 3 options | None | Sect. 9 | 7.41 | 59.53 | 3.42 | 31.74 |
| Unigram | 3 options | None | Sect. 1-9 | 7.12 | 61.11 | 3.35 | 33.13 |
| Unigram | 3 options | None | Sect. 5-9 | 7.11 | 61.17 | 3.34 | 33.33 |
| Trigram | 3 options | None | Sect. 5-9 | 7.06 | 61.44 | 3.32 | 33.73 |
| Trigram | Best case | 2 way | Sect. 5-9 | 6.75 | 63.13 | 2.91 | 41.92 |

Character Sequence Transformation. This step is implemented as a probabilistic string edit process. The confusion tables for edit operations are estimated using Viterbi style training on $\langle O, C \rangle$ pairs in training data. Our current implementation allows for substitution, deletion, and insertion errors, and does not use context characters.

Final Cleanup. At this stage, special symbols that were inserted into the character sequence are removed and the final output sequence is formed. For instance, segment boundary symbols are removed or replaced with spaces depending on the language.

3.2 Decoding

Decoding is the process of finding the “best” W for an observed (\hat{O}, \hat{b}) , namely

$$\hat{W} = \operatorname{argmax}_W \{ \max_{a,C} [P(\hat{O}, \hat{b} | a, C, W) P(a | C, W) P(C | W) P(W)] \} \quad (5)$$

Decoding within the FSM framework is straightforward: we simply compose all the components of the model in order, and then invert the resulting FSM. This produces a single transducer that takes a sequence of OCR characters as input, and returns all possible sequences of truth words as output, along with their weights. The most probable sequence among those returned by the composition can be used as the output of the post-OCR correction process. Alternatively, the resulting lattice or N -best list can easily be integrated with other probabilistic models over words.

4 Experimental Evaluation

Although most researchers are interested in improving the results of OCR on degraded documents, we are primarily interested in developing and improving OCR in new languages for use in NLP. A possible approach to retargeting OCR for a new language is to employ an existing OCR system from a “nearby” language, and then to apply our error correction framework. For these experiments, therefore, we created our experimental data by scanning a hardcopy Bible using both an English and a French OCR system. (See Kanungo et al. [4] and Resnik et al. [9] for discussion of the Bible as a resource for multilingual OCR and NLP.) We have used the output of the

English system run on French input to simulate the situation where available resources of one language are used to acquire resources in another language that is similar.

It was necessary to pre-process the data in order to eliminate the differences between the on-line version that we used as the ground truth and the hardcopy, such as footnotes, glossary, cross-references, page numbers, etc. We have not corrected hyphenations, case differences, etc.

Our evaluation metrics for OCR performance are Word Error Rate (WER) and Character Error Rate (CER), which are defined as follows. The results reported are obtained by ignoring character case.

$$\text{WER}(W_{truth}, W_{OCR}) = \frac{\text{WordEditDistance}(W_{truth}, W_{OCR})}{|W_{truth}|} \quad (6)$$

$$\text{CER}(C, O) = \frac{\text{CharEditDistance}(C, O)}{|C|} \quad (7)$$

Since we are interested in recovering the original word sequence rather than the character sequence, evaluations are performed on lowercased and tokenized data. Note, however, our system works on the original case OCR data, and generates a sequence of word ids, that are converted to a lowercase character sequence for evaluation.

We have divided the data, which has 29317 lines, into 10 equal size disjoint sets, and used the first 9 as the training data, and the first 500 lines of the last one as the test data. The WER and CER for the English OCR system on the French the test data were 18.31% and 5.01% respectively. The numbers drop to 17.21% and 4.28% when single character tokens and tokens with no alphabetical characters are ignored. The WER and CER for the output generated on French by the French OCR system were 5.98% and 2.11%.

4.1 Reduction of OCR Error Rates

We evaluated the performance of our model by studying the reduction in WER and CER after correction. The input to the system was original case, tokenized OCR output, and the output of the system was a sequence of word ids that are converted to lowercase character sequences for evaluation.

Table 2: WER, CER, and reduction rates ignoring single characters and non-alphabetical tokens

| Conditions | | | | Results | | | |
|---------------------|-----------|-------|-----------|---------|----------|---------|----------|
| LM | WC | SG | EM | WER (%) | Red. (%) | CER (%) | Red. (%) |
| Original OCR Output | | | | 17.21 | - | 4.28 | - |
| Unigram | 3 options | None | Sect. 9 | 3.97 | 76.93 | 1.68 | 60.75 |
| Unigram | 3 options | None | Sect. 1-9 | 3.62 | 78.97 | 1.60 | 62.62 |
| Unigram | 3 options | None | Sect. 5-9 | 3.61 | 79.02 | 1.58 | 63.08 |
| Trigram | 3 options | None | Sect. 5-9 | 3.52 | 79.55 | 1.56 | 63.55 |
| Trigram | Best case | 2 way | Sect. 5-9 | 3.15 | 81.70 | 1.14 | 73.36 |

All the results are summarized in Table 1. The conditions side gives various parameters for each experiment. The language model (LM) is either (word) unigram or trigram. Word to character conversion (WC) can allow 3 case variations that are mentioned before, or simply pick the most probable one for each word. Segmentation (SG) can be disabled, or 2-way split and merges may be allowed. Finally, the character level error model (EM) may be trained on various subsets of training data. Table 2 gives the adjusted results when ignoring all single characters and tokens that do not contain any alphabetical character.

As can be seen from the tables, as we increase the training size of character error model from 1 section to 5 sections, the performance increases. However there is a slight decrease in performance when the training size is increased to 9 sections. This suggests that our training procedures, while effective, may require refinement as additional training data becomes available. When we replace the unigram language model with a trigram one, the results improve as expected. However, the most interesting case is the last experiment where 2-way word merge/split errors are allowed.

Word merge/split errors cause an exponential increase in the search space; when there are n words that needs to be corrected together, there are N^n possible combinations where N is the vocabulary size. We imposed various restrictions on the model to reduce the search space and achieve acceptable execution times. Despite the imposed restrictions, the ability to handle word merge/split errors improves performance significantly.

5 Related Work

There has been considerable research on automatically correcting words in text in general, and correction of OCR output in particular. Kukich [10] provides a general survey of the research in the area. Unfortunately, there is no commonly used evaluation base for OCR error correction, making comparison of experimental results difficult.

Some systems integrate the post-processor with the actual character recognizer to allow interaction between the two. In an early study, Hanson et al. [11] reports a word error rate of about 2% and a reject rate of 1%, without a dictionary. Sinha and Prasada [12] achieves 97% word

recognition, ignoring punctuation, using an augmented dictionary, a Viterbi style algorithm, and manual heuristics.

Many systems treat OCR as a black box, generally employing word and/or character level n -grams along with character confusion probabilities. Srihari et al. [13] is one typical example and reports up to 87% error correction on artificial data, and relying (as we do) on a lexicon for correction. Goshtasby and Ehrlich [14] presents a method that based on probabilistic relaxation labeling, using context characters to constrain the probability of each character. They do not use a lexicon but do require the probabilities assigned to individual characters by the OCR system.

Jones et al. [15] describe an OCR post-processing system comparable to ours, and reports error reductions of 70-90%. Their system is designed around a stratified algorithm. The first phase performs isolated word correction using rewrite rules, allowing words that are not in the lexicon. The second phase attempts correcting word split errors, and the last phase uses word bigram probabilities to improve correction. In comparison to our work, the main difference is our focus on an end-to-end generative model versus their stratified algorithm centered around correction.

Pal et al. [16] describes a method for OCR error correction of an inflectional Indian language using morphological parsing, and reports correcting 84% of the words with a single character error. Although it is limited to single errors, the systems demonstrates the possibility of correcting OCR errors in morphologically rich languages.

Although segmentation errors have been addressed to some degree in previous work, to the best of our knowledge our model is the first that explicitly incorporates segmentation. Similarly, many systems make use of a language model, a character confusion model, etc., but none have developed an end-to-end model that formally describes the OCR process from the generation of the true word sequence to the output of the OCR system in a manner that allows for statistical parameter estimation. Our model is also the first to explicitly model the conversion of a sequence of words into a character sequence.

6 Conclusions and Future Work

We have presented a flexible, modular, probabilistic generative OCR model designed specifically for ease of integration with probabilistic models of the sort commonly found in recent NLP work, and for rapid retargeting of OCR and NLP technology to new languages.

In a rigorous evaluation of post-OCR error correction on real data, illustrating a scenario where a black-box commercial English OCR system is retargeted to work with French data, we obtained a 70% reduction in word error rate over the English-on-French baseline, with a resulting word accuracy of 97%. It is worth noting that our post-OCR correction of the English OCR on French text led to better performance than a commercial French OCR system run on the same text.

We are currently working on improving the correction performance of the system, and extending our error model implementation to include character context and allow for character merge/split errors. We also intend to relax the requirement of having a word list, so that the model handles valid word errors.

Finally, we plan to challenge our model with other languages, starting with Arabic, Turkish, and Chinese. Arabic and Turkish have phonetic alphabets, but also pose the problem of rich morphology. Chinese will require more work due to the size of its alphabet. We are optimistic that the power and flexibility of our modeling framework will allow us to develop the necessary techniques for these languages, as well as many others.

Acknowledgements

This research was supported in part by National Science Foundation grant EIA0130422, Department of Defense contract RD-02-5700, DARPA/ITO Cooperative Agreement N660010028910, and Mitre agreement 010418-7712.

We are grateful to Mohri et al. for the AT&T FSM Toolkit, Clarkson and Rosenfeld for CMU-Cambridge Toolkit, and David Doermann for providing the OCR output and useful discussion.

References

- [1] Jim Krane. New handheld translators could be used in Iraq war. *The Honolulu Advertiser*, 2002. October 7.
- [2] W. B. Croft, S. M. Harding, K. Taghva, and J. Borsack. An evaluation of information retrieval accuracy with simulated OCR output. In *Symposium of Document Analysis and Information Retrieval, ISRI-UNLV*, 1994.
- [3] David Doermann. The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding: CVIU*, 70(3):287–298, 1998.
- [4] Tapas Kanungo, Philip Resnik, Song Mao, Doewan Kim, and Qigong Zheng. The bible, truth, and multilingual optical character recognition. in revision.
- [5] David Doermann, Huanfeng Ma, Burcu Karagöl-Ayan, and Douglas W. Oard. Translation lexicon acquisition from bilingual dictionaries. In *Ninth SPIE Symposium on Document Recognition and Retrieval*, San Jose, CA, 2002.
- [6] Okan Kolak, William Byrne, and Philip Resnik. A generative probabilistic OCR model for NLP applications. In *Human Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Alberta, Canada, May 2003. To appear.
- [7] Mehryar Mohri, Fernando C. N. Pereira, and Michael Riley. A rational design for a weighted finite-state transducer library. *Lecture Notes in Computer Science*, 1436, 1998.
- [8] Philip Clarkson and Ronald Rosenfeld. Statistical language modeling using the CMU-Cambridge Toolkit. In *ESCA Eurospeech*, 1997.
- [9] Philip Resnik, Mari Broman Olsen, and Mona Diab. The Bible as a parallel corpus: Annotating the ‘Book of 2000 Tongues’. *Computers and the Humanities*, 33:129–153, 1999.
- [10] Karen Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, December 1992.
- [11] Allen R. Hanson, Edward M. Riseman, and Edward G. Fisher. Context in word recognition. *Pattern Recognition*, 8:33–45, 1976.
- [12] R. M. K. Sinha and Biendra Prasada. Visual text recognition through contextual processing. *Pattern Recognition*, 21(5):463–479, 1988.
- [13] Sargur N. Srihari, Jonathan J. Hull, and Ramesh Choudhari. Integrating diverse knowledge sources in text recognition. *ACM Transactions on Office Information Systems*, 1(1):68–87, January 1983.
- [14] Ardeshir Goshtasby and Roger W. Ehrich. Contextual word recognition using probabilistic relaxation labeling. *Pattern Recognition*, 21(5):455–462, 1988.
- [15] Mark A. Jones, Guy A. Story, and Bruce W. Ballard. Integrating multiple knowledge sources in a bayesian OCR post-processor. In *IDCAR-91*, pages 925–933, St. Malo, France, 1991.
- [16] U. Pal, P. K. Kundu, and B. B. Chaudhuri. OCR error correction of an inflectional indian language using morphological parsing. *Journal of Information Science and Engineering*, 16(6):903–922, November 2000.

Government

OCR for Collection, Management, and Retrieval of Documents: Development and Trial of a Documentation Exploitation Suite

Luis Hernandez Christian Schlesiger

Army Research Laboratory
2800 Powder Mill Rd, Adelphi, MD 20783-1197

Abstract

The Information Age has paved the way for the growing ability to print and publish material that is relevant to people's everyday needs. In remote areas of the world, printing varies widely in quality due to differences in printing techniques and paper type. These differences pose a particular challenge for the U.S. Army, who at a moment's notice may get deployed anywhere in the world. Once deployed, these troops must deal with foreign documents in vast number and, for efficiency, must consider automated means for processing them: from collecting and tracking documents, translating them, and processing audit trails, to archiving, and ultimately retrieving the information stored in them to aid decision-making. The Army Research Laboratory, in conjunction with other military organizations, steered an effort to create a prototype system that automates these processes and that, moreover, performs them on documents that vary in printed quality. This document exploitation (or DocEx) suite integrates optical character recognition (OCR), machine translation (MT), document management, workflow, and indexing, as well as retrieval of information by topic or keyword. This suite underwent trial by a working Army unit using Arabic, Russian, and Spanish documents representative of those found in the field. This paper describes lessons learned during the integration and user trial of this prototype. In addition, insight is provided into future needs and areas of potential research.

1 Introduction: The need for DocEx

In today's global environment, information is easy to come by. One can go any place in the world and find information in magazines, newspapers, office documents, newsletters, or letters. The content is typically drafted in the native language of the author.

When the army deploys to new areas of the world, it needs to access collected information in a timely manner. This information can have a great impact on the decisions the army makes daily.

In times past, the process of document exploitation (DocEx) – that is, collecting documents, organizing and

translating them, and then exploiting them for critical content – was all done by hand. But as the number of documents increases, the ability to deal with the information in a timely manner declines. Important information may not be found in time to affect a decision. Faced with a deluge of information, the army realizes the need to automate aspects of handling documents.

An automated process for DocEx would need to be able to handle all those tasks that the manual process includes. It must be able to organize and report on the information collected in a manner similar to current practice in order to ease the transition to army users.

1.1 OCR and Document Quality

Not all regions of the world are at the same level of capability when it comes to producing documents. In the small office setting, one can expect a wide disparity of printing technology ranging from sophisticated laser printers to dot matrix printers to even manual typewriters. In larger businesses, mass-printing technology for newspapers and magazines differs almost as widely. The nature of the paper affects document quality. Many documents are printed using recycled or onion skin-like paper. In several countries, the bulk of the printed material displays defects such as bleed through and faint print. It is clear that for army deployments, where an automated process will be used, it must be capable of working with documents across the spectrum of quality.

For printed material, this means that an automated DocEx process must include OCR technology and that OCR must be able to (a) handle documents of low quality and variable noise; and (b) process documents generated in the native tongue of the region.

1.2 Machine Translation

Most typically, non-linguists are the collection agents processing documents that are found in the field [1]. Similarly, non-linguists and linguists alike will want to examine those documents for needed information. The large quantity of collected documents poses a formidable challenge – how to process and extract

relevant information in a timely fashion. Thus, the need for a machine translation (MT) capability for an army DocEx suite is clear. Automated queries made on a database of collected documents are most often conducted in English, so an army DocEx suite also needs cross-language information retrieval capabilities.

1.3 Organization and Tracking

As documents are collected, they must be tracked and organized. The volume of documents expected in a typical collection run demands that a sophisticated archiving file-system be integral to automated DocEx.

Meta-data about documents acquired must also be maintained: where the document came from and how it was found can be as important as the content itself. Entering meta-data about a document is an integral part of the process that must be captured.

As the document moves through the automated process, modules will make changes and create new versions of it. For example, a scanned piece of paper will produce an image of the document. OCR will produce a text file of that scanned image. MT will produce a translated text file. Human translators can even create a separate version in another text file. All of these files need to be kept as part of the document and maintained for possible exploitation. Additionally, keeping archived copies of the document at every stage will facilitate version control.

It is important to know who did what to a document throughout its lifetime in the collection. Audit trails and quality assurance steps added to the process will also provide a high level of security and database integrity.

1.4 Retrieving Information

A final function in automated DocEx is information retrieval and extraction. As documents are collected and entered into document management, a database or archive will be created. If sufficiently organized, this database can be queried to provide the information needed for the decisions army users will need to make.

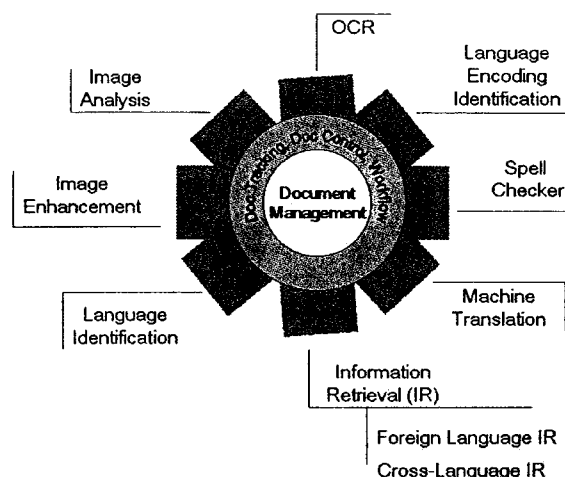
Typically, users interested in retrieving relevant information search documents based on a given set of questions. These questions are commonly referred to as Priority Information Requirements (PIR). Often, non-linguists create the PIRs in English. Through the application of OCR and MT, documents are available as approximate English translations for searching. This provides analysts with a means to retrieve information. The search results can be displayed in several ways: original image, OCR text, machine translated text, and human translation when available. The retrieved documents can then be browsed in English or in the native tongue if the analyst is a linguist. As PIRs change rapidly, the automated process provides the ability to search any new documents as well as existing information stored in the database.

2 DocEx Design Goals

The objective of the DocEx project reported here is to incorporate the processes described above by integrating a suite of GOTS/COTS information processing and foreign language translation tools in a web-based, stand-alone environment. Such an integrated suite of tools aims to enhance tactical operations and support national level foreign document collection, management, translation, and dissemination in relevant scenarios.

The various technologies available for working with documents (both text and images) interact through a document management system that allows them to work together and to contribute to a single, searchable repository. This concept, as depicted in Figure 1, is the cornerstone in an overarching concept of operations

Figure 1: Concept Diagram for Document Exploitation Tactical Support Suite.



3 Development: FALCon as a starting point

The Army Research Lab (ARL) develops applications to demonstrate and test the utility of tools to process natural language. One application, FALCon (Forward Area Language Converter), is an end-to-end system that links OCR of paper input in various languages with machine translation (MT) of those languages into English [1, 2]. The FALCon process works on a single page at a time under the assumption that a non-linguist can perform triage tasks on documents he acquires in the field by assigning them varying levels of priority based on what he finds in the rough translation.

The FALCon system provides an obvious starting point for the development of a Document Exploitation automated suite. FALCon already integrates two of the

relevant technologies needed in the automated process: OCR and MT software. FALCon has been extensively field tested and reviewed, and has gone through several iterations that refined and expanded its language coverage. These iterations also added technologies such as language and text file encoding identification, conversion, and file format handling. The capability that FALCon provides to handle and preset the entire process from beginning to end could serve as the model for the DocEx workflow. The core pieces of FALCon could be extracted and then built into the DocEx automated process easily and quickly.

3.1 Development: Commercial Products

To complete the DocEx process flow, it was evident that document management, tracking, and information retrieval functionality was necessary to automatically process large quantities of documents. FALCon was not designed for this purpose.

Thought was put into creating from scratch the missing pieces for the automated DocEx. However, a search done on the commercial market produced some interesting alternatives that already had the capability desired. There was no need to reinvent the wheel if existing technology could be married with the core FALCon components.

Vredenburg, a commercial company in the area local to ARL, makes a product called HighView. This software package is a workflow and document management system that has been in the commercial world for many years. It has gone through several version iterations and has been deployed in commercial and government facilities. This posed a reasonable alternative to provide the desired functionality within a sixty-five day rapid application development timeframe.

In addition, when HighView is used with an Oracle database, one can leverage Oracle's entity and topic extraction capability. This feature, only available in English, allows the user to query all the documents in the collection by subject as well as by keywords. Using the system, an analyst can find documents that have the exact words, relevant subject, or similar words, as well as documents that are simply like a sample block of text.

3.2 Putting it together

To develop the automated DocEx system, the core HighView product was used as the centerpiece. It provided a core environment that allowed developers to create modules and templates to interact in a sophisticated workflow process.

New modules were created in the workflow for OCR and MT technology from FALCon. Also taken from FALCon were the lessons learned necessary to develop coupling modules for handling multilingual document

types, text encoding, and image type conversions. These lessons come from our experience resolving mismatched file types due to integrating multiple COTS products across categories of OCR and MT in an embedded process.

To facilitate retrieval, it was necessary to capture information about each input document and associate this information to the output of the process.

Developed as new technology within the DocEx workflow was the capability of entering in document meta-data as a document entered the collection. This process utilized existing tracking features that would be familiar to army users.

To support information retrieval needs, new modules were developed that allowed users to enter in PIR information and to make advanced searches on the collection.

MT and OCR are conducted automatically by the system as a document is scanned or an electronic file submitted. These processes occur moments after a document is entered, making the information available, in English, for query.

At all steps of the automated process, there is the opportunity for quality and assurance by a human. Every document generated by every module is kept together with the originals for reference. Documents can be checked in and out by users and changes made with a complete history for back reference. At every stage, this information is available for query.

Because documents encountered in the field are often low quality, the OCR output can be inaccurate, so access to images as well as text is important in retrieval.

4 Demonstrating the Capability

The integration of the DocEx automated process was a rapid development completed in sixty-five days. The goal was to demonstrate the concept to army users already conducting manual document exploitation.

As a prototype for demonstration, only three languages were incorporated for OCR and MT: Russian, Spanish, and Arabic. The prototype was taken to an army unit in Europe.

5 Results

A week was given to demonstrate the system. Users were first shown the capabilities of the automated process and led through guided exercises. They were trained and then left on their own to process foreign language documents in the three languages. They were given the chance to observe the OCR and MT in action and to make queries on the collection that was built during the week.

Documents used during this demonstration were selected for content, and were a combination of low

quality book selections and high quality pages laser printed material.

At the end of the demonstration period, the users were given a questionnaire designed to assess the DocEx Suite in three categories: effectiveness, suitability, and usability. The question responses were in the form a choice of strongly agree (5) to strongly disagree (1).

Five users were polled with the questionnaire. Their response in all categories averaged above 4 (agree). Their comments were very positive, and many expressed a desire to have the system in their hands immediately. The responses of the users can be seen in Tables 1, 2, and 3 for each of the categories of questions. The questionnaire is included in the Appendix A. When the user did not respond to the question, a zero was entered into the table, but not used in the average of the response.

Table 1: User Responses to Effectiveness Category

| Question | Respondents | | | | | Avg |
|----------|-------------|---|---|---|---|------|
| | 1 | 2 | 3 | 4 | 5 | |
| 1 | 5 | 4 | 5 | 0 | 5 | 4.75 |
| 2 | 5 | 4 | 5 | 4 | 5 | 4.6 |
| 3 | 5 | 4 | 5 | 4 | 5 | 4.6 |
| 4 | 5 | 4 | 5 | 5 | 2 | 4.2 |
| 5 | 5 | 4 | 0 | 4 | 3 | 4 |
| 8 | 5 | 4 | 0 | 4 | 5 | 4.5 |
| 9 | 5 | 4 | 4 | 4 | 5 | 4.4 |
| 10 | 5 | 4 | 4 | 4 | 4 | 4.2 |
| 12 | 5 | 4 | 4 | 4 | 5 | 4.4 |
| 13 | 5 | 4 | 4 | 5 | 5 | 4.6 |
| 14 | 5 | 5 | 2 | 4 | 4 | 4 |
| 15 | 5 | 4 | 0 | 4 | 5 | 4.5 |

Table 2: User Responses to Suitability Category

| Question | Respondents | | | | | Avg |
|----------|-------------|---|---|---|---|------|
| | 1 | 2 | 3 | 4 | 5 | |
| 1 | 5 | 4 | 4 | 4 | 5 | 4.4 |
| 2 | 0 | 4 | 4 | 5 | 5 | 4.5 |
| 3 | 5 | 4 | 4 | 5 | 5 | 4.6 |
| 4 | 5 | 4 | 0 | 5 | 5 | 4.75 |
| 5 | 5 | 4 | 4 | 4 | 5 | 4.4 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 |

Table 3: User Responses to Usability Category

| Question | Respondents | | | | | Avg |
|----------|-------------|---|---|---|---|------|
| | 1 | 2 | 3 | 4 | 5 | |
| 1 | 5 | 4 | 4 | 5 | 5 | 4.6 |
| 2 | 5 | 4 | 4 | 4 | 5 | 4.4 |
| 3 | 5 | 2 | 4 | 3 | 4 | 3.6 |
| 4 | 5 | 4 | 4 | 4 | 5 | 4.4 |
| 5 | 5 | 4 | 4 | 4 | 0 | 4.25 |

6 Future needs

As a result of users' favorable response to the first prototype, work is being pushed ahead for a phase-2 prototype. This prototype will incorporate more languages, necessitating evaluation and integration of more OCR and MT technology.

OCR of low quality documents was a bottleneck to the translation process, as it is in FALCon [3]. OCR software generally works very well with high quality images of documents using very common fonts in languages with an extensive history of OCR research. Variation of any of these factors can substantially impact OCR accuracy. The OCR output is a cause for concern because it forms the input for machine translation and will correspondingly impact retrieval.

We seek new technology for image cleanup to insert into the workflow to enhance the OCR and improve subsequent processes.

References

- [1] Fisher, F., Voss, C.: FALCon, an MT System Support Tool for Non-linguists. *Proceedings of the Advanced Information Processing and Analysis Conference*, McLean VA (1997)
- [2] Holland, M., Schlesiger, C.: High-Mobility Machine Translation for a Battlefield Environment. *Proceedings of NATO/RTO Systems Concepts and Integration Symposium*, Monterey, CA. Hull, Canada: CCG, Inc. (ISBN 92-837-1006-1) (1998) 15/1-3.
- [3] Holland, M., Schlesiger, C., Hernandez, L.: What System Developers Need to Select OCR for Authentic Tasks: Evaluating End-to-End Systems. *Proceedings 2001 Symposium on Document Image Understanding Technology*. Columbia, MD (2001)

Appendix A

The following is the questionnaire given at the end of the demonstration of the DocEx suite. Most questions

were followed by multiple choice selection: Strongly Agree, Agree, Somewhat Agree, Do not Agree, Strongly Disagree. Users were encouraged to make remarks after each question.

EFFECTIVENESS

1. *The DOCEX SUITE was adequate for simultaneous multiple user access.*
2. *The DOCEX SUITE was adequate to inventory information on batches of documents/media.*
3. *The DOCEX SUITE was adequate to inventory information on individual documents/media.*
4. *The DOCEX SUITE was adequate for keyword indexing.*
5. *The DOCEX SUITE was adequate for exploitation potential/tracking.*
6. *DOCEX SUITE - the amount of information I had to enter manually was:*

__Too Much __Just enough _Enough __Not enough
7. *DOCEX SUITE failed to provide a means for entering the following information that I thought necessary.*
8. *The DOCEX SUITE significantly enhances my capability to perform the mission/operations (as intelligence collector, screener, analyst, mission manager).*
9. *DOCEX SUITE was adequate for collecting data to be exported to the National HARMONY System.*
10. *DOCEX SUITE data entry and query capabilities were adequate and allowed for duplication checking.*
11. *I would like to see the following capabilities added to DOCEX SUITE.*
12. *DOCEX SUITE is an effective tool for disseminating screening data and translations to the analysts for validation and reporting.*
13. *DOCEX SUITE is an effecting means to*

get translated information into the reporting chain for dissemination throughout the chain of command.

14. *The DOCEX SUITE tools for reviewing text-processing operations (OCR, Machine Translation, Full Text Indexing, etc.) are adequate.*

15. *If I had to go on a mission tomorrow, I would want to be able to use DOCEX SUITE.*

DOCEX SUITE SUITABILITY

1. *DOCEX SUITE system speed is adequate to support data management of the digital images of documents.*
2. *The DOCEX SUITE web browser is suitable for accessing information.*
3. *The DOCEX SUITE is suitable for storage of document information for both inventory and identification purposes.*
4. *The DOCEX SUITE is an effective tool for collecting, translating, reporting, analyzing, disseminating, and receiving feedback to reported information.*
5. *The DOCEX SUITE screen navigation is sufficient for rapid access of material.*
6. *The DOCEX SUITE is easy to troubleshoot. (SysAdmin Only).*

DOCEX SUITE USABILITY

1. *It is easy to access information with DOCEX SUITE.*
2. *The DOCEX SUITE is easy to learn.*
3. *The training I received on the DOCEX SUITE was adequate for this demonstration.*
4. *The Oracle Knowledge Base was easy to understand and use.*
5. *The Workflow monitoring tools (Composer Monitor) were adequate.*

6. I would like to have more training in the following areas prior to using DOCEX SUITE.

7. What did you like most about using the DOCEX SUITE?

8. What did you like least about using the DOCEX SUITE?

Overall Comments:

MOS Rank

Are you a Linguist? Yes No

Proficiency Level:

If yes, what language(s):

Are you an Analyst? Yes No **Computer**

Efficiency Level: High Med Low

Transitioning Experimental HMM OCR: From Lab to Field

Christian Schlesiger

Luis Hernandez

Michael Lee

Army Research Laboratory

2800 Powder Mill Rd, Adelphi, MD 20783-1197

Abstract

As part of its mission, the army may be deployed at a moment's notice to any part of the world. In these areas, there is a need to know the content of paper documents written in a foreign language as they are encountered. The army has made use of end-to-end automatic translation systems that couple optical character recognition (OCR) with machine translation (MT) to make quick assessments of these documents. However, often the availability of OCR in low diffusion or less commonly taught languages is either very small or prohibitively expensive since commercial companies develop OCR software based on sales and demand of the market. The army has a growing need for an OCR product that can be developed and deployed very rapidly in end-to-end automatic translation systems for these languages the commercial world has little interest in. In this paper we will consider an HMM OCR product for Arabic and Simplified Chinese that was developed with this need in mind. It was evaluated and integrated into the Forward Area Language Converter (FALCon) [1] developed by ARL and then deployed to a number of army users for field use.

1 Introduction

In an ever-changing world, today's army has been deployed rapidly to remote areas where the language spoken is not English. When linguists are not available, the army is faced with few alternatives when encountering documents in the field in these foreign languages. One alternative is to make use of automated software systems that can OCR and translate documents to give a rough assessment of the document's content for triage and prioritization. One such system is FALCon [1, 2], developed by ARL that links OCR and MT software technology for a variety of languages.

The problem becomes further compounded when the foreign language is not usually taught to linguists. One reason for this may be that the language is considered a low diffusion or density language. The reliance on automated systems increases in these areas for this lack of available linguists. However, the ability to develop automated systems in these languages is hampered by the low availability of OCR and MT technology for the same reasons.

In developing FALCon, ARL makes use of

commercial software packages whenever possible. The commercial market for OCR is driven by sales and the size of the market. Companies will not invest in a product that will not sell. Or, if they do develop such a product it is priced prohibitively for large deployments. Thus, the range of choices for OCR in less common obscure languages is quite narrow.

For some languages, there is the option of finding an OCR developed within the country of the language's origin. However, these present development and logistical issues when integrating into an army system for English speaking users. Often the documentation, graphical interface, application development information, and technical support are in the foreign language as well.

One solution is to develop an OCR system that can be taught to recognize any language. The chief advantage of such a system is that the core OCR capability can be trained in a very short amount of time provided a large enough sample set is collected. This rapid development time is very attractive for army system development. This technology could be integrated into Falcon, and then as new languages are brought to the forefront of interest, a set of sample documents could be collected and then used to train the OCR.

2 BBN Byblos OCR Development

In conjunction with other government agencies who have a similar need for rapid deployment OCR, ARL collaborated with BBN Technologies to develop this type of OCR product. This OCR would utilize a recognition engine that made use of Hidden Markov Model technology that was used successfully in a variety of speech recognition engines.

As a concept to demonstrate the feasibility of this approach, two languages were chosen to train the engine: Arabic and Simplified Chinese. Document corpora for these languages was easy to obtain and in ample supply.

3 Evaluation and Integration

At the end of the development period, the first prototype demonstration package from BBN was examined for integration within FALCon. Two questions needed to be answered: How well did the

OCR perform in terms of recognition accuracy in the two languages? How difficult will it be to integrate the experimental OCR into FALCon?

3.1 OCR Accuracy

Before the OCR could be integrated into the end-to-end system, it was evaluated for performance. We performed a very small pilot test to get a rough idea of the accuracy of the OCR software.

We took a sample set of ten Arabic and ten Simplified Chinese documents of varying length and ran them through the OCR. None of the documents in the sample set were longer than one full page of text, or shorter than a quarter page. All of the documents were on high quality laser-printed paper. The OCR output was compared character-by-character to the original source files. The results can be seen in Table 1.

Table 1: Pilot test of BBN OCR character accuracy and process time

| Source File | Arabic | | Chinese | |
|----------------|--------------------|-------------------|--------------------|-------------------|
| | Process Time (sec) | Chars correct (%) | Process Time (sec) | Chars correct (%) |
| 1 | 40 | 73.27 | 706 | 74.59 |
| 2 | 28 | 73.27 | 694 | 74.30 |
| 3 | 26 | 76.21 | 726 | 72.67 |
| 4 | 35 | 76.01 | 782 | 67.97 |
| 5 | 36 | 72.50 | 880 | 66.37 |
| 6 | 31 | 74.68 | 801 | 69.46 |
| 7 | 31 | 74.57 | 774 | 69.14 |
| 8 | 33 | 74.29 | 509 | 64.29 |
| 9 | 37 | 75.12 | 550 | 70.40 |
| 10 | 40 | 70.81 | 547 | 69.48 |
| Average | 33.7 | 74.07 | 626.5 | 69.87 |

Early in the tests it was found that the performance time of the OCR was significantly longer than the commercial products that we had worked with previously. We used a simple stopwatch to manually time the OCR process. We knew that our users would be interested in this data, as it would impact performance.

The percentage of characters correct is much lower than one would expect. The training data used to create the OCR was not laser printed. It was much lower quality, and the HMM engine is sensitive to this difference in the test data.

The process time, we also learned, is directly proportional to the number of characters in the

language. Chinese, which has thousands of characters for the engine to consider, took a much longer time to recognize. Improvements in code optimization would probably reduce this time considerably. However, there will still be a time trade off for languages with many characters that may require more research to alleviate.

3.2 Integration and Issues with Falcon

The demonstration package that was provided for our experiment was not productized. It was a conceptual piece of software meant to prove the feasibility of the approach. However, it was felt to be a good test for it to be integrated into the full end-to-end FALCon system so users could take a look at it.

The lack of productization presented many challenges for integration. The OCR software was not optimized for speed, required the installation of subprograms such as Perl, did not have a full API, accepted only one type of image file input, produced only one type of text output, and had a barebones GUI designed only to demonstrate the product.

In order to do a quick and seamless integration for evaluation with FALCon we had to solve and work around most of these limitations. We wanted FALCon users to be unaware that the OCR they were using was different from the commercial products already integrated except by performance only.

Working with BBN Technologies to meet these challenges resulted in several patches that made the integration much easier. Most of the patches concerned the format and file types for input and output to the OCR software.

The designers originally output Arabic text in the presentation form of Arabic Unicode. This turned out to be incompatible with the Arabic MT package already integrated into FALCon. It was necessary to change the output to the more general base-form Arabic Unicode text.

The demonstration package only accepted TIF format files for image input. We integrated a freeware image conversion utility to expand the types of image files that could be sent through the OCR by converting them when necessary to TIF.

Installation issues also arose during the integration process. FALCon was made to be a distributable software package via CDROM. We needed to develop an installation script for the OCR package that would place all the files in the right place, including a critical font, set permissions correctly for several folders, and allowed the installation of either or both of the languages available.

Other limitations were identified during the integration process. The OCR can only process fairly simple forms of text formatting. Columns, tables, and other complex formats caused the OCR to perform very poorly, or even crash outright.

As mentioned earlier, documents that differ in quality greatly from those used to train the OCR impact its performance. Our tests with laser printed material were one end of the spectrum. Documents that are very dirty and of low quality also fall into this category. However, this is true of commercial OCR as well.

4 Ongoing and Future Work

FALCon with the experimental OCR was completed and sent out to users. ARL continues to collect data on how it performs on real world documents.

In the lab, we've learned that the OCR performance improves if image-preprocessing routines are run on images prior to recognition. Even very simple routines for despeckling and skew correction boost performance several percent. We are interested in finding more tools that could increase performance by cleaning up the document image.

Completing the product process on the OCR would also be worthwhile. A robust product would be less prone to crashing when given an image of poor quality. It would also perform much faster when the algorithms are optimized for speed.

Several more languages are being trained using this OCR engine. As they are completed, they will be integrated into FALCon and sent to the field. This rapid development and deployment of OCR will expand the usability of end-to-end OCR and MT processes.

References

- [1] Fisher, F., Voss, C.: Falcon, an MT System Support Tool for Non-linguists. *Proceedings of the Advanced Information Processing and Analysis Conference*, McLean VA (1997)
- [2] Holland, M., Schlesiger, C.: High-Mobility Machine Translation for a Battlefield Environment. *Proceedings of NATO/RTO Systems Concepts and Integration Symposium*, Monterey, CA. Hull, Canada: CCG, Inc. (ISBN 92-837-1006-1) (1998) 15/1-3.
- [3] Holland, M., Schlesiger, C., Hernandez, L.: What System Developers Need to Select OCR for Authentic Tasks: Evaluating End-to-End Systems. *Proceedings 2001 Symposium on Document Image Understanding Technology*. Columbia, MD (2001)
- [4] DeHart, J., Schlesiger, C.: Issues in Optical Character Recognition for Army Machine Translation. *Proceedings 1999 Symposium on Document Image Understanding*. Annapolis, MD (1999)
- [5] Bazzi, I., Natarajan, P., Schwartz, R., Kornai, A., Lu, Z., Makhoul, J.: OCR of Degraded Documents using HMM-Based Techniques.

Proceedings 1999 Symposium on Document Image Understanding. Annapolis, MD (1999)

- [6] Natarajan, P., Schwartz R., Makhoul, J.: OCR of Low-resolution Text Images from Diverse Sources. *Proceedings 2001 Symposium on Document Image Understanding Technology*. Columbia, MD (2001)
- [7] Natarajan, P., Lu, Z., Schwartz R., Bazzi, I., Makhoul, J.: Multilingual Machine Printed OCR. *International Journal of Pattern Recognition and Artificial Intelligence*. Vol. 15, No. 1 (2001) 43-63
- [8] Bazzi, I., Schwartz R., Makhoul, J.: An Omnifont Open-Vocabulary OCR System for English and Arabic. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 21, No. 6, June 1999.



Automatic Content Extraction (ACE)

The Effects of Document Analysis on Automatic Content Extraction

Jonathan K. Davis
ACE Program Manager

Knowledge Discovery from Text Division
Department of Defense
Fort George G. Meade, MD

jkdavis@afterlife.ncsc.mil

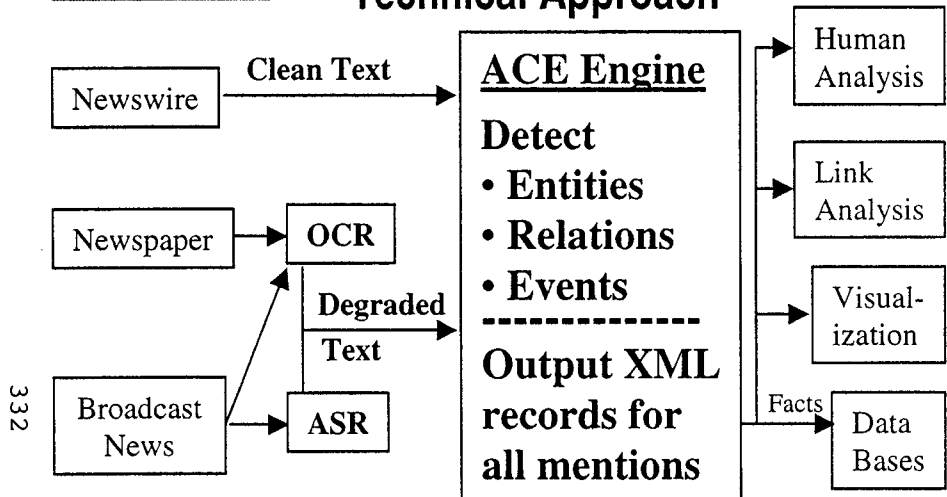


Automatic Content Extraction (ACE)

Department of Defense / J.K. Davis



Technical Approach



Objectives and Performance Goals

Objectives

- Extract Info from Texts of Varying Quality
- Detect Unique Entities & Relations
 - **Collect All Mentions by Entity/Relation**
- Track Entities Within & Across Documents
- Output XML for Follow-On Processes

Performance Goals

- Extract 95% of the Value in Document

Accomplishments

- Researched Entity Detection Algorithms
 - Recognize Each Mention of an Entity
 - Resolve All Mentions to Proper Entity
- Researched Relation Detection Algorithms
 - Recognize Explicit & Implicit Relations
- Performed 4 Metrics-Based Evaluations
 - Used Value-based Measure
 - Correct, Miss, Mis-Typed

Programmatics

- ACE Pilot Study (Mar 99 - Aug 00)
 - Definition of Entity Detection Task
- ACE Phase 2 (Oct 00 - Dec 02)
 - Entity and Relation Detection
 - Evaluations for ACE/TIDES/EELD
- ACE Phase 3 (Oct 03 - Sep 05)
 - Add Event Detection
 - Become Multi-Lingual (EN/CH/AR)



Automatic Content Extraction



- Core Research: Natural Language Processing of Text
 - Entity Detection Task
 - Coreference Resolution
 - Accumulate All Mentions (Proper, Common, Pronominal)
 - Metonymy Resolution
 - Determine “**Intended**” Entity of “**Literal**” Mention
 - “**The White House** said ...”: **ORG** which resides in **FAC**
 - Relation Detection and Characterization
 - Explicit Mention and Implicit Inference
 - Temporal Attributes (a la TIDES TIMEX2)
 - Absolute, Relative, and Unanchored Times
 - Proposition Representation for Relations and Events
 - Model Relations/Events via Predicate Argument Structure
 - Cross-Document Entity Tracking
 - Form Equivalence Class for the Same Globally-Unique Entity
 - Model Textual and Topical Contexts



Example of Extracted Entities



“On Saturday night, the **Glass House** dance club in **Pomona** was packed with **800 people** enjoying the holidays when the **local police** and members of a **Los Angeles County hazardous materials team** walked in and stopped the music ... A few minutes earlier **someone** had called **911** and reported what has become a common threat ... **The caller** said the **lethal bacteria anthrax** had been released in **the club**.”

<PERSON ID=P1>
Desc=“800 people
enjoying the holidays ”
</PERSON>

<PERSON ID=P2>
Desc=“members of a Los
Angeles County hazardous
materials team”
</PERSON>

<PERSON ID=P3>
Desc=“someone”,
“the caller”
</PERSON>

<GPE ID=G1>
Name=“Pomona”
</GPE>

<GPE ID=G2>
Name=“Los Angeles County”
</GPE>

<FACILITY ID=F1>
Name=“the Glass House”
Desc=“dance club”, “the club”
</FACILITY>

<SUBSTANCE ID=S1>
Name=“anthrax”
Desc=“the lethal bacteria”
</SUBSTANCE>

<ORGANIZATION ID=O1>
Desc=“the local police ”
</ORGANIZATION>

<ORGANIZATION ID=O2>
Desc=“a Los Angeles County
hazardous materials team”
</ORGANIZATION>

<ORGANIZATION ID=O3>
Name=“911 ”
</ORGANIZATION>



Example of Extracted Relations



“On Saturday night, the Glass House dance club in Pomona was packed with 800 people enjoying the holidays when the local police and members of a Los Angeles County hazardous materials team walked in and stopped the music ... A few minutes earlier someone had called 911 and reported what has become a common threat ... The caller said the lethal bacteria anthrax had been released in the club.”

335

<AT ID=R1>

What=<FACILITY ID=F1>

Where=<GPE ID=G1>

</AT>

<AT ID=R2>

What=<PERSON ID=P1>

Where=<FACILITY ID=F1>

When=“Saturday night

</AT>

<AT ID=R3>

What=<ORG ID=O1>

Where=<FACILITY ID=F1>

When=“Saturday night

</AT>

<AT ID=R4>

What=<PERSON ID=P2>

Where=<FACILITY ID=F1>

When=“Saturday night

</AT>

<ROLE ID=R5>

Arg1=<PERSON ID=P2>

Arg2 =<ORG ID=O2>

Role=“member”

</ROLE>

<PART_OF ID=R6>

Parent=<GPE ID=G2>

Child =<ORG ID=O2>

</PART_OF>

<AT ID=R7>

What=<SUBSTANCE ID=S1>

Where=<FACILITY ID=F1>

When=“Saturday night

</AT>



Automatic Content Extraction Program Products



- ACE Development and Evaluation Corpora
 - Example Extractions (Answer Keys) for EDT and RDC
 - corpus:
 - <http://morph ldc.upenn.edu/Projects/ACE2/>
 - <http://www.nist.gov/speech/tests/ace/phase2/resources/>
 - guidelines:
 - <http://morph ldc.upenn.edu/Projects/ACE2/>
 - Alembic Workbench for Creating Answer Keys
 - <http://www.mitre.org/resources/centers/it/g063/nl-index.html>
 - Evaluation Software
 - <http://www.nist.gov/speech/tests/ace/phase2/resources/>
- Proposition Banks for English and Chinese
 - <http://www.cis.upenn.edu/~ace/>



Automatic Content Extraction Synergy with Image Analysis



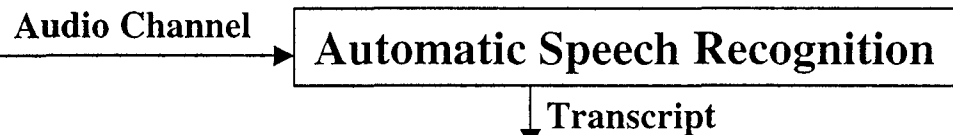
- Mutual Assistance in Document Analysis
 - ACE and Image Analysis Systems Can Provide Hints or Validation of Each Others' Assessments
 - ACE Systems Produce Time-Aligned, Structured Data from Text
 - Detection of Selected Types of Entities, Relations, & Events
 - Extraction of Text Strings Mentioning These Objects
 - Output of Mentions by Unique Object in XML Format
 - Image Analysis Systems Produce Metadata about Image
 - Text Discovered in the Image
 - Information on Objects Found in the Image
 - Inter-System Communication of Analyzed Data
 - Sharing of Insights Leads to Better Analysis
 - ACE Metadata Enhances Automatic Indexing of Images



Automatic Content Extraction Synergy with Image Analysis

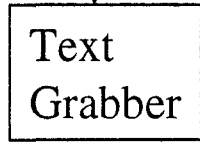
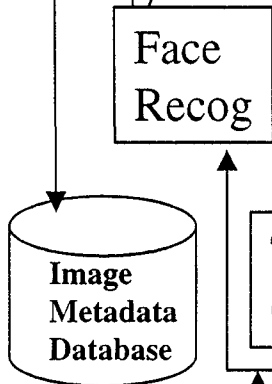


- Mutual Metadata Enhances Entity & Text Detection
 - Recognition & ID of Entities in Images & Text
 - Validation of Spelling in Text & Image



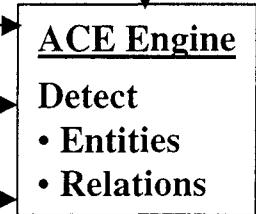
Today in UN Security Council, United States Secretary of State Colin Powell pressed the American case for possible war to disarm Iraq. Accompanied by CIA Director George Tenet, Powell presented ...

Closed Caption Text

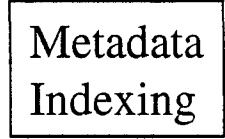


Plaque Info: "United States"

Face Recognition: "Colin Powell", George Tenet"



ORG1="UN ", ORG2 ="UN Security Council", GPE1=("United States", "American")
 PER1=("Colin Powell", "Powell"), GPE2="Iraq", ORG3="CIA", PER2="George Tenet",
 AT=(What=PER1, Where=ORG1), AT=(What=PER2, Where=ORG1),
 ROLE=(Arg1= PER1, Arg2= GPE1, Role="Secretary of State")
 ROLE=(Arg1= PER2, Arg2= ORG3, Role="Director")



An Automation Tool For the Detection of Sensitive Information

Gary DeWitt
Information System Security Officer
Office of Classified and Controlled Information Review
Office of Security
U.S. Department of Energy

Abstract

The Office of Classified and Controlled Information Review (OCCIR) in the U.S. Department of Energy ensures that all documents prepared at DOE Headquarters are properly marked to identify protected information they contain (if any) and that all declassified documents the federal government prepares for public release contain no DOE sensitive information requiring protection under applicable statutes, regulations, and Executive orders.

To meet the challenge of Executive Order 12958, OCCIR has invested in automation tools to assist its document review staff in reviewing the literally millions of pages of historical documents. The automated review of historical documents poses a significant challenge in that the quality of the documents, in terms of readability by automated systems, is often poor, thus converting to electronic format for processing by automated systems is problematical. This problem is exacerbated with microfilm because reproducing the document onto microfilm leads to further degradation of quality. Although the exact number of microfilm reels that OCCIR must review is not fully known, most estimates place it in the tens of thousands of reels.

Over the past year, the Information Technology Program in OCCIR has developed a technology to assist in the review of microfilm. The system, known as the Microfilm Information Review System (MIRS), is composed of three distinct subsystems: a scanner to convert images on microfilm to electronic image format; an OCR engine to convert the electronic image format to electronic machine readable text; and a software knowledge base through which the electronic text is filtered through to search for indicators of sensitive information.

This talk will describe each of the three subsystems, discuss the optimal settings for scanning and optical character recognition, and describe the process by which the knowledge base works. Results of blind tests with the MIRS will be shown and sample output will be included.

The Declassification Challenge: Can Technology Make a Difference?

Richard Warshaw
Chief, CIA Declassification Center

In April 1995, there was a watershed in U.S. Government declassification with the signing by then President Clinton of Executive Order 12958, which contained the so-called "automatic declassification provision." This provision provided nine exemptions and mandated the automatic declassification of nonexempt records of historical value 25 years old or older on April 2000. This original date was extended to 2003 and, at this writing a further extension to 2006 appears to be in the cards. The Executive Order, as intended, set off a massive declassification effort Government-wide to systematically review and exempt still-sensitive records in whole or in part where permitted from automatic declassification. While still difficult to estimate, over a billion (not million!) pages of historical records require review. While the majority of these records are being reviewed manually on a pass/fail basis, many tens of millions are being reviewed and redacted (declassified-in-part) using electronic systems; this is particularly true in intelligence agencies where many records retain at least partially their sensitivity and would otherwise be judged exempt in full on a pure pass/fail basis.


This presentation will deal with the potential and reality of the application of technology to declassification. With eight years of experience in high-volume declassification, we have a far better idea of what technological tools have contributed and could possibly contribute to ensuring that high quality reviews are completed in an efficient and timely manner. It is clear that the human reviewer will always be key to this process but tools many believe can mitigate errors and reduce costly re-reviews, and allow a limited workforce to deal with a massive workload. While significantly increasing the speed of review is possible, the challenge is to increase speed while maintaining review quality. Review errors potentially jeopardize still sensitive national security information, even more so in the post-9/11 environment, and undermine confidence in the review process. Some progress has been made in the tools arena, but with our increasing understanding of the process, and availability of many new and relevant technologies, there is a tantalizing possibility for real breakthroughs.

The speaker is Chief of the CIA Declassification Center and has held that position since the signing of the automatic declassification Executive Order in 1995. He also chairs the interagency working group on automatic declassification involving representatives of more than 20 declassification programs. He is an engineer by training and brings to the task a strong belief in the importance of the declassification mission and the role that new technology or borrowed technology can play in ensuring successful implementation of automatic declassification and other major Government declassification programs both statutory and voluntary.

VACE


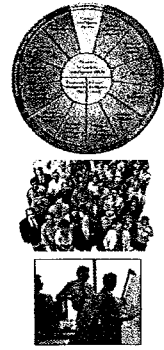
(Video Analysis & Content Extraction)

Advanced R&D Program




Dr. John D. Prange
 VACE Program Manager
 JPrange@nsa.gov
<http://www.ic-arda.org>
 301-688-7092

How ARDA Interacts

- **Community organizations**
 - Plans, forecasts, oversight
 - Customer champions
- **Thrust panels / managers**
 - R&D problem statements
 - Internal peer review
- **Industry and academia**
 - Principal funding recipients
 - External peer review and staff


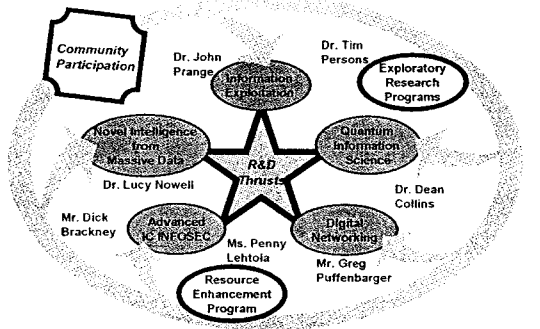
Introducing ARDA



A joint Department of Defense / Intelligence Community organization launched in Dec 98

- **MISSION:**
 - Incubate revolutionary R&D for the shared benefit of the Intelligence Community
- **MEANS:**
 - A nimble, cross-community organization
 - A modest, yet significant budget
 - Small, outward-looking staff working as "honest brokers" and "agent provocateurs"


Current ARDA Programs

R&D Thrust

- **Community Participation** (Dr. John Prange)
- **Information Exploitation** (Dr. Tim Persons)
- **Exploratory Research Programs**
- **Quantum Information Science** (Dr. Dean Collins)
- **Digital Networking** (Mr. Greg Puffenberger)
- **Resource Enhancement Program** (Ms. Penny Lehtola)
- **Advanced IC INFOSEC** (Mr. Dick Brackney)
- **Novel Intelligence from Massive Data** (Dr. Lucy Nowell)


What ARDA Does



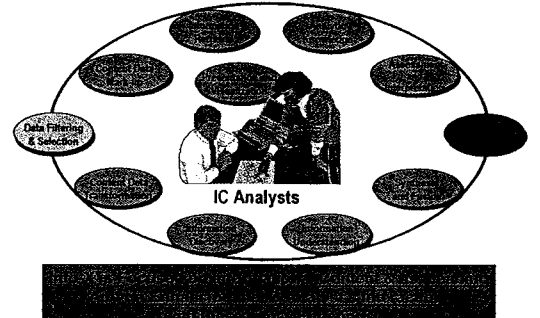
We originate and manage R&D programs

- With fundamental impact on future operational needs and strategies
- That demand substantial, long-term venture investment to spur risk-taking
- That progress measurably toward mid-term and final goals
- That take many forms and employ many delivery vehicles

Information Exploitation (Info-X)



What Functions Does it Include?

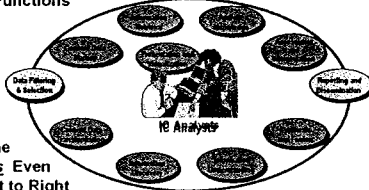


IC Analysts

- Data Filtering & Selection
- Data Analysis
- Data Correlation
- Data Integration
- Data Mining
- Data Presentation
- Data Storage
- Data Transfer
- Data Validation

ARDA Observations

- Analyst is at the Center of Info-X and is Directly Involved in virtually Every Info-X function
- In particular, tool implementations of Info-X functions need to be guided by analysts' evolving mental creations, understandings and knowledge throughout a tightly coupled interactive process
- No Order to Analytic Functions
- Required Analytic Intellect Increases Significantly Left to Right
- Current Capabilities for Automated Machine Processing Decreases Even More Dramatically Left to Right



7

ARDA Video Exploitation
Why is it such a huge problem?

- Multiple Data Format Factors**
 - Variety of Compressed/Uncompressed Formats/Encodings
 - Degree of lossy compression
 - Environment, Lighting, Time of day
 - Panning, Zooming, Fixed, Mounted on moving object
 - Resolution
 - Single vs. multiple, & overlapping vs. non-overlapping, etc.

10

ARDA Current Info-X R&D Programs

- AQUAINT**
Advanced QUESion & Answering for INTelligence
- GI²Vis**
Geospatial Intelligence Information Visualization
- NDHB**
Non-Linear Dynamics from Human Behavior
- LEMUR**
Statistical Language Modeling for Information Retrieval

Full R&D Programs consisting of Multiple Phases

Exploratory R&D Programs consisting of Programs 1-Year + Option Year

8

ARDA Video Exploitation
Why is it such a huge problem?

- Large Video Collections**
 - Volumes of Video continue to grow exponentially
 - Strains storage, network distribution, and data management systems
 - Really bad news: Current volumes of Video already exceed analytic capacity

11

ARDA Video Exploitation
Why is it such a huge problem?

- Multiple Video Types**
 - Foreign Broadcast News (e.g. FBIS)
 - Fixed Surveillance Videos - Maybe multiple overlapping cameras
 - Indoor/Outdoor scenes from moving cameras
 - Business Meeting/Conference Videos
 - UAV-type Videos

9

ARDA Video Exploitation
Why is it such a huge problem?

- Commercial Technology**
 - Focus on High Quality Video Data Management (DM)
 - DM is very important to the IC but it is clearly not enough
- Lack of robust software tools that allow content based access to video data means that:**
 - Analyst typically access video sequentially; Very inefficient
 - Human intervention is required to annotate video for indexing purposes
 - Content based routing based on automated processing is lacking
 - Flexible Ad Hoc Search and Browsing tools do not exist

12

VACE
Video Analysis & Content Extraction

ARDA

- Envisioned as a high risk, long term R&D Program:
 - Phase I Fall 2000 - Winter 2002
 - Phase II Summer 2003 - Summer 2005
 - Phase III Fall 2005 - Fall 2007
- VACE Program Committee:
 - CIA - NSA - DIA - NIMA - ARDA
- Research Goals
 - Robust person, vehicle, and text detection and recognition.
 - Cross media content analysis and extraction
 - Fully automatic video indexing based in image, text, and audio content.
 - Efficient methods for representing video content.
 - Event Detection, Recognition and Understanding.
 - Low cost video corpus marking and preparation

13

VACE Phase 1:
Research Objectives

ARDA

| | |
|--|---|
| <p>5. Multi-modal Fusion</p> <ul style="list-style-type: none"> • Combine multiple sources of information to support detection, recognition and understanding for given scenes. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • What can be understood from combining processing results from speech, text, and image in a video of a Middle east political rally? | <p>6. Event Understanding</p> <ul style="list-style-type: none"> • Form inferences from occurrence or re-occurrence of activity. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • Person A just left the scene with Person B's briefcase. • The yellow truck outside the building is the same one circling the building all week. |
|--|---|

16

VACE Phase 1:
Research Objectives

ARDA

| | |
|--|--|
| <p>1. Object Detection</p> <ul style="list-style-type: none"> • Robust and accurate detection, location and counts of an object: <ul style="list-style-type: none"> - Face in a room. - Text in a scene. - Vehicle in a scene. - Region where the object appears. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • How many persons in the scene? • Find the next scene where text/people appear. | <p>2. Object Recognition</p> <ul style="list-style-type: none"> • Determine the specific instance of an object class. <ul style="list-style-type: none"> - Person - Vehicles - Text <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • Does Mr. Prime Minister appear at the news conference? • Is Mr. Terrorist at the airport? |
|--|--|

14

VACE Phase 1:
Research Objectives

ARDA

| | |
|--|--|
| <p>7. Video Summary</p> <ul style="list-style-type: none"> • Methods to reduce information representation • Scenario based activity summarization. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • Rapid browsing • Mr. Terrorist and Mr. Violent met in the Rome airport last evening. • Mr. Negotiator agreed with Ms. Economic Advisor on a framework for partnership. | <p>8. Video Query by Example</p> <ul style="list-style-type: none"> • Retrieve video sequences like the example sequence. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • Is this scene duplicated in the database? • Find all of the news broadcasts that show this video sequence of a rocket launch. |
|--|--|

17

VACE Phase 1:
Research Objectives

ARDA

| | |
|---|---|
| <p>3. Object Tracking</p> <ul style="list-style-type: none"> • Determine the path of a known object within a sequence. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • What areas of the airport were visited by Mr. Terrorist? | <p>4. Motion Analysis</p> <ul style="list-style-type: none"> • Quantify the movement of objects or phenomena in a video sequence. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • Is that really Mr. Terrorist by his manner of walking (e.g. his gait)? • What can we learn from the gestures and facial expressions of Ms. Prime Minister? |
|---|---|

15

VACE Phase 1:
Research Objectives

ARDA

| | |
|--|---|
| <p>9. Multi-Modal Video Mining</p> <ul style="list-style-type: none"> • Automatically discovering trends, patterns and associations in video. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • Mr. Terrorist appears in Rome when Mr. Violent arrives with a briefcase. • When Ms. Prime Minister appears, Mr. Negotiator is always present. | <p>10. Kinematics Analysis</p> <ul style="list-style-type: none"> • Identify an object, event or phenomenon by its motion. <p>TECHNOLOGY</p> <hr/> <p>CAPABILITY</p> <ul style="list-style-type: none"> • Does Mr. Terrorist's car contain all of his equipment based on its acceleration? |
|--|---|

18

VACE Phase 1:
Research Objectives

11. Integrity Analysis

- Examine video data for signs of manipulation or tampering.

TECHNOLOGY
CAPABILITY

- Has this video been altered?
- Is that really Mr. Terrorist talking to Mr. Leader?

12. Video Mensuration

- Measure temporal, spatial, or spectral dimensions in data

TECHNOLOGY
CAPABILITY

- What is the position of the Yellow Truck in the surveillance video relative to the government facility? How accurately can you measure this position?

19

Object Tracking & Event Understanding
University of Southern California
(Co-PI: Ram Nevatia & Gerard Medioni)

22

VACE Phase 1:
Research Objectives

13. Model Reconstruction

- Construct an accurate 3D geometric model of an object from one or more video streams.

TECHNOLOGY
CAPABILITY

- Based on this set of video tapes, is the existing model of Mr. Terrorist's compound accurate?

20

Goals

- Infer interesting events in a video**
 - Some examples: people meeting, exchanging objects, gesturing....
 - Events may take place over a range of time scales
 - Requires object recognition
- Provide a convenient form to define events**
 - An *event recognition language (ERL)* that can be compiled automatically to produce recognition programs
- Compute a structured representation of video**
 - Events and objects
 - Spatial and temporal relations between them

23

VACE Program Contractors

Boeing
SRJ
Univ. of Southern California
HNC Software
Carnegie Mellon Univ. (Informedia)
Carnegie Mellon Univ. (Robotics Inst.)
GE Corp Research Division
NGC, TASC
Brown Univ.
Princeton Univ.
Univ. of Maryland

21

Tracking Moving Blobs

- Blobs split and merge due to occlusion, similarity with background, noise blobs appear...**
- Use of region similarities and trajectory smoothness can help infer good trajectories**
- Two approaches**
 - Multi-resolution spatio-temporal grouping
 - Perceptual grouping using graph representation and tensor voting

24

USC

Some Tracking Problems

• Input: A set of moving regions

• Output: Identification and Tracking of the objects

25

USC

Multiple Objects Tracking

28

USC

Tracking Demonstration

26

USC

Tracking with multiple cameras

- **Motivation**
 - Multiple cameras can provide larger field of view and reduce effects of occlusion
- **Issues**
 - Registration of multiple views
 - Calibration of different cameras
 - Synchronization of different video streams
 - Different frame rates
 - Grouping of trajectories across views

29

USC

Tracking by Tensor Voting

- Propagate motion and shape information into uncertainty regions around paths in the graph
- Accumulate votes by tensor voting process
- Includes process for merging and splitting of blobs to yield smooth trajectories

27

USC

Two Video Streams

Two partially overlapping views

30

Projective Registration USC

ARDA

Registered Views

Registered Trajectories

31

Example of Multi-thread Event USC

ARDA

34

Summary: Blob Tracking USC

ARDA

- Real-time tracking with hierarchical approach and graph-based methods
 - Region similarity exploited but not continuity
- Perceptual grouping using tensor voting
 - Slower, but better tracks
- Merging multiple camera tracks is in early development stage
- Future work
 - Improve efficiency of perceptual grouping
 - Develop adaptive multi-scale approach
 - Integrated detection and tracking from multiple cameras

32

Annotated “object_transfer” USC

ARDA

35

Multi-thread Event Modeling USC

ARDA

- Global activities can be described by several actors performing related actions
 - Action threads may overlap in time but are related by temporal/logical constraints
- Represented by an event graph
 - Nodes are single-thread events
 - Links indicate temporal relations represented by *Interval-Based Temporal Logic*
 - Only qualitative relations between intervals such as “before”, “after”, “during”, “overlap”, ... are specified


33

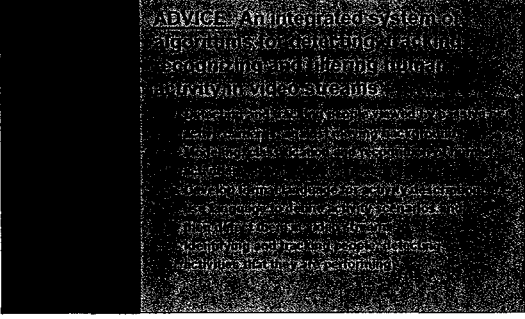
Summary: Accomplishments USC

ARDA

- Object detection and tracking
 - Good progress for mobile objects including cases with some
 - Inference of 3-D trajectories
 - Integration of views from multiple cameras
- Generic framework for event recognition
 - Defined an event hierarchy, ERL and procedure for its compilation
 - Developed efficient methods for recognizing events


36

Activity Detection by Video Content Estimation 
 ARDA University of Maryland (PI: Larry Davis)

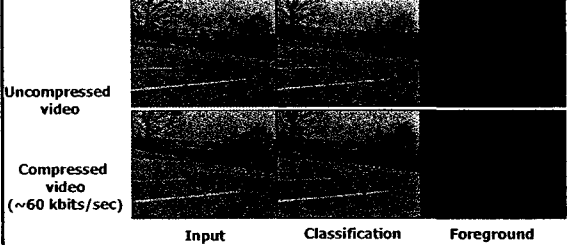


ADVICE: An integrated system of algorithms for detecting, tracking, and segmenting foreground objects in video sequences.

37

Problem 
 ARDA

- Prior methods do not work well with compressed video.




Uncompressed video

Compressed video (~60 kbits/sec)


Input Classification Foreground

40

UMD Program Overview 
 ARDA


- Low level techniques
 - Background Modeling and Detection (Davis)
 - Tracking and segmentation (Elgammal)
 - Motion Segmentation (Aloimonos)
- Human Identification
 - Utilizing Color-Spatial Information for Object Tracking and Person Identification (Davis)
 - Probabilistic Face Recognition from Compressed Imagery (Chellappa)
- Event modeling and recognition
 - Space-Time Grouping (DeMenthon)
 - Activity Recognition (Doermann)

38

Methodology 
 ARDA


- “Vector” quantization of sample background values into codebooks
 - A compressed form of background model.
 - Allows utilization of long training periods with limited memory
- Adaptively model dynamic background over long time periods using a layered model.
- Separate brightness from chromaticity to handle illumination variations.

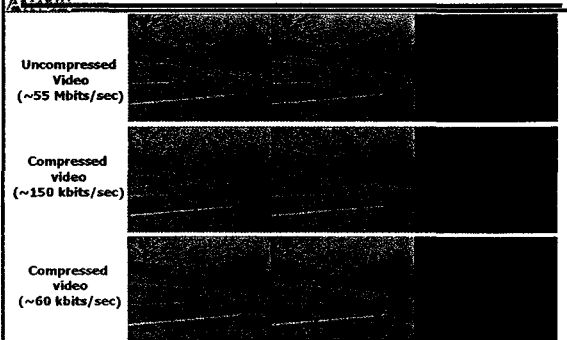
41

Background modeling and detection 
 ARDA

- Improved methods for background modeling and segmentation
- Methods for parameter estimation
 - Global and local illumination changes (shadows and highlights)
 - Variable (moving) backgrounds
 - Video quality
 - Presence of foreground objects in background during training

39

Results 
 ARDA



Uncompressed Video (~55 Mbits/sec)

Compressed video (~150 kbits/sec)

Compressed video (~60 kbits/sec)

Input Classification Foreground

42

ARDA **Videos** UMD

43

ARDA **Tracking with occlusion** UMD

46

ARDA **Segmentation of Groups into Individuals** UMD

- Objective: Build appearance representations of people when they are isolated that enables segmentation of foreground regions when they occlude one another

44

ARDA **Multi-Modal Content Extraction in Video** Carnegie Mellon University (PI: Takeo Kanade)

47

ARDA **Representation** UMD

- Model the person as a vertical set of blobs.
- Each blob has a color distribution that is independent of the position within the blob

45

ARDA **CMU Program Overview** CMU

- Scalable object detection (Kanade, Schneidermann)
 - Face Detection that is robust to variations in pose, lighting, compression, and image quality
 - Vehicle Detection for multiple makes/models (e.g., passenger cars, trucks, and military vehicles)
- Tracking and Recognition (Chen)
- Pose estimation and enhancement (Chen)

48

CMU

Object Detection Research Goals

- Face Detection – robust to variations in pose, lighting, compression, and image quality
- Vehicle Detection – multi-kinds (e.g., passenger cars, trucks, and military vehicles)

49

CMU

Classification

| Input Window | Parts | Probabilities | Decision Rule |
|--------------|-------|---------------|---------------|
| | | 0.674 | Face |
| | | 0.987 | |
| | | · | |
| | | · | |
| | | 0.541 | |

• Sample exhaustively with overlay

52

CMU

Face Detection Car Detection

50

CMU

Training images of object

Sampled parts

$P(\text{part}, x, y | \text{object})$

Training images of "non-object"

Sampled parts

$P(\text{part}, x, y | \text{non-object})$

53

CMU

Scalable Object Detection

Input Window

Local Region and Position

$x = 3$
 $y = 4$

Dimensionality Reduction & Quantization

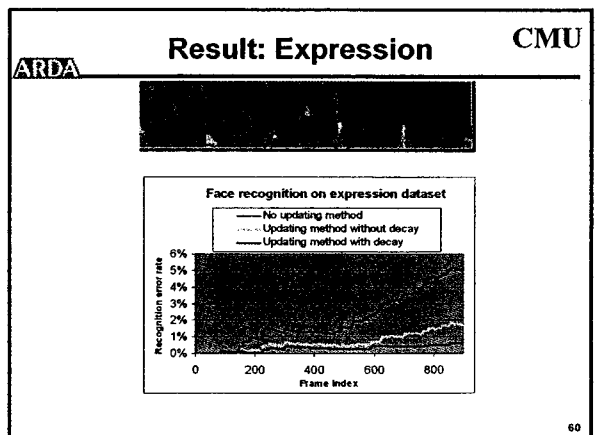
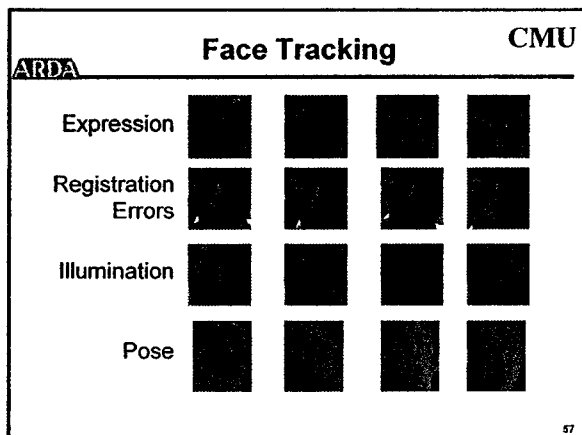
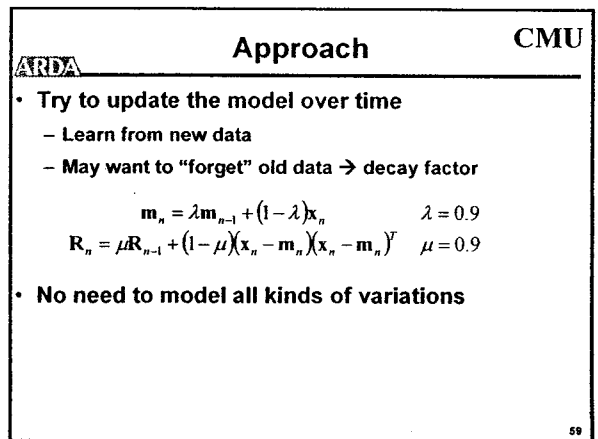
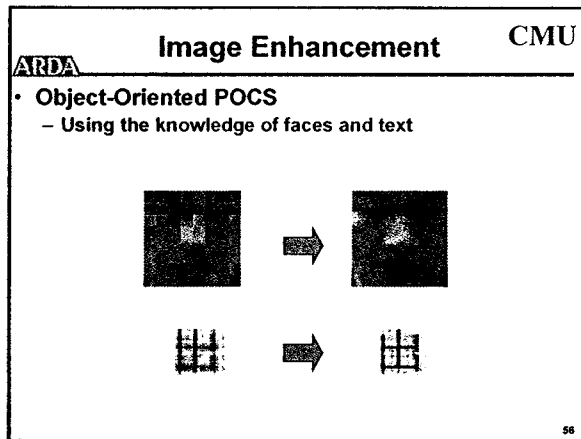
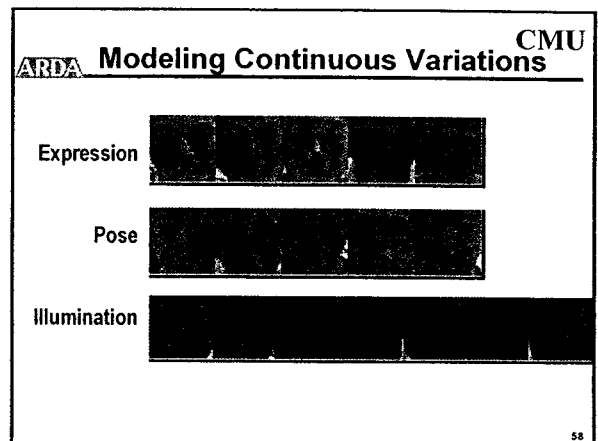
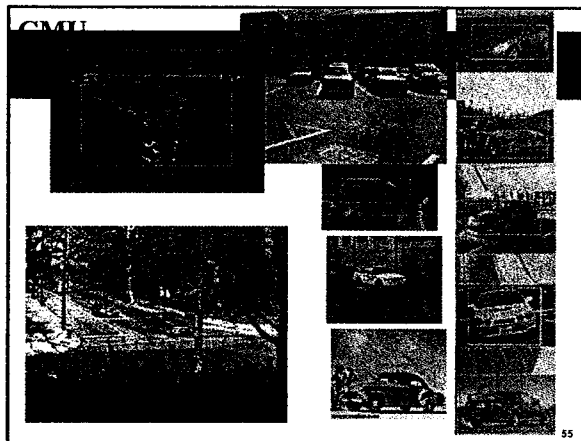
Part and Position

part = 00981
 $x = 3$
 $y = 4$

51

CMU

54



CMU

ARDA VACE Phase I Research Results

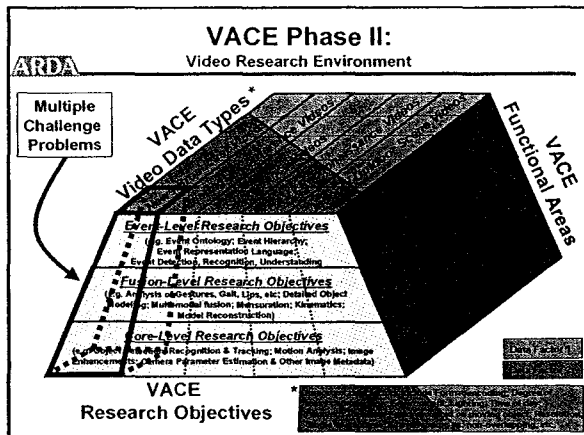
- Developed robust capabilities for person detection and tracking in compressed video streams.
- Developed robust capabilities for the automated detection of objects in single frame images.
- Developed new normalization techniques to improve recognition in video data.

61

ARDA Video Data: VACE Phase II Plans

- Actively pursuing multiple sources of video data to support VACE Phase II R&D Program; In particular:
 - Planning to acquire or generate video data to support the Phase II Challenge Problems
 - Planning to annotate video data to support both metric based component level and application level Phase II sponsored evaluations

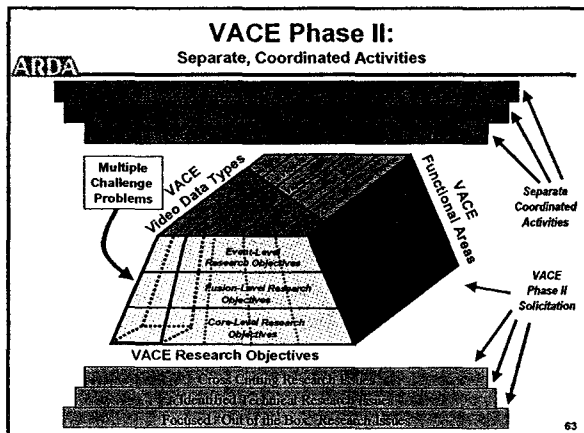
64



ARDA VACE Phase II: User Testbed / System Integration

- Pull together best available system components emerging from VACE Program research efforts
 - Couple VACE components with existing GOTS and COTS software
- Develop end-to-end VACE prototype(s) aimed at specific Operational Video Exploitation environments
- Government-led effort:
 - Directly Linked into Sponsoring Agency's Technology Insertion Organizations
 - Close, working relationship with working Analysts
 - Provide external system development support
 - TBD External Organization will be identified by the VACE Program to orchestrate/implement System Integration / Testbed efforts
 - May also utilize additional external researchers as Consultants / Advisors in this effort

65



ARDA Measures of Success

- Robust Video Indexing
 - Reliable, Value Added
 - Robust Content Representation
 - Automatic Event Understanding
- Quantitative Performance Breakthroughs
 - Open evaluations using representative corpora
 - Significant advances against baselines
- Enable New Applications
 - Efficient Video Retrieval, Intelligent Browsing, Data Mining, Link Analysis, Visualization
- Vastly Improved Application Performance
 - Users' Evaluation in operational context

66

Performance Evaluation: Why

- Understand Strengths and Weaknesses of Various Alternatives
- Track/Document Significant Progress
- Understand Potential Trade-offs
- Facilitate Cooperation, Integration
- Predict Performance on Unseen Data
- This is not a contest
- Absolute Numbers are Less Important
 - Range of useful values depending upon task

67

Discussion / Questions ?

70

VACE Phase I Evaluations

- Performed Detection Evaluations for:
 - Text
 - Face
 - Human bodies
 - Vehicles
- Initial attempt at text tracking evaluation
- Evaluations separated by about four months

68

VACE Phase II Evaluations

How will VACE Phase II evaluations be different from VACE Phase I evaluations?

- Move from detection/tracking to object/event recognition
- Specific evaluation tasks and schedule will be defined by Phase II efforts
- Revisit prior evaluations periodically
- Evaluate on larger data sets
- More participants

69

Author Index

-A-

Ablavsky, Vitaly 135
Alam, H...... 265

-B-

Baird, Henry 17
Beckley, Russell 41
Beitzel, Steven..... 145
Borsack, Julie 153
Breuel, Thomas..... 17, 209, 245, 293
Byrne, William 313

-C-

Cantwell, Ken 37
Coombs, Jeff..... 41
Cumbee, Carson..... 239

-D-

Darwish, Kareem 181
Davis, J.K. 331
Decerbo, Michael..... 47
D'Errico, Tom..... 307
DeWitt, Gary..... 339
Doermann, David..... 53, 97, 309
Drayer, Tom..... 37
Droettboom, Michael 275

-E, F-

Fancy, Joanna..... 289
Femiani, John 287
Fleet, David 121
Ford, Glenn 199
Fujinaga, Ichiro..... 275

-G-

Golebiowski, Lynn..... 219
Govindaraju, V...... 189
Grossman, David 145

-H-

Hartone, R. 265
Hernandez, Luis 321, 327
Hauser, Susan..... 171
Henderson, Thomas 253

-I, J-

Jensen, Eric 145
Jing, Hongyan..... 111

-K-

Karagol-Ayan, Burcu 53
Katsnelson, Yuliya..... 197
Keller, Tom..... 47
Kolak, Okan..... 313
Kompalli, S. 189

-L-

Larner, Daniel 121
Lee, Michael 327
Lee, Sangjik 67
Li, Huiping..... 97
Lopresti, Daniel 17, 111

-M-

Ma, Huanfeng..... 53
MacMillan, Karl 275
Mahoney, James..... 121

Makhoul, John 47
 Manmatha, R..... 77
 Mokrzycki, Gregory.....291
 Myers, Gregory.....259

-N-

Nartker, Tom.....153
 Natarajan, Prem 47

-O-

Oard, Douglas181

-P-

Pollak, Joshua.....135
 Popat, Kris..... 17
 Prange, John.....343
 Pereda, Ray 41
 Price, Robert..... 87

-Q, R-

Rahman, Fuad.....265
 Rath, T.M..... 77
 Razdan, Anshuman.....287
 Resnik, Philip.....313
 Rowe, Jeremy.....287

-S-

Sabir, Tehseen.....171
 Sadeh, Mohammad..... 41
 Sakakihara, Alan.....229
 Sarkar, Prateek.....17, 245
 Saund, Eric121
 Setlur, S.189
 Schlesiger, Christian..... 321, 327
 Schwartz, Rich 47
 Shi, Zhixin..... 67
 Shih, Chilin.....111
 Shilman, Michael 93
 Shin, Yong-Chul..... 67

Simard, Patrice 11
 Snorrason, Magnus135
 Srihari, Sargur 67
 Stevens, Mark.....135
 Summers, Kristen 159, 197
 Swaminathan, Lavanya253

-T-

Tagjva, Kazem41, 153
 Tarnikova, T.....265
 Tjahjadi, T.265
 Thoma, George 171, 199
 Tomai, Catalin..... 67
 Turner, Mark..... 197, 295

-U, V-

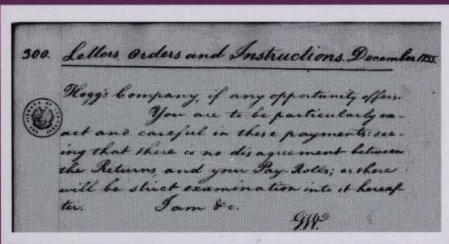
Vemulapati, R.189

-W-

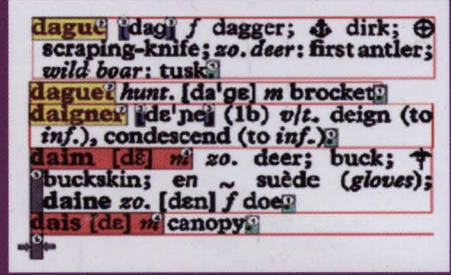
Warshaw, Richard.....341
 Watts, Gabriel..... 13
 Wilcox, C.265
 Wnek, Janusz..... 31

-X, Y, Z-

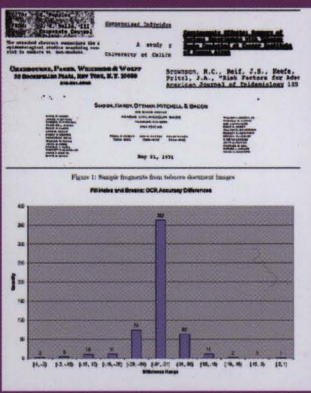
Young, Ron..... 41
 Zhang, Bin 67
 Zheng, Yefeng 97
 Zi, Gang.....309
 Zukasa, Anthony..... 87



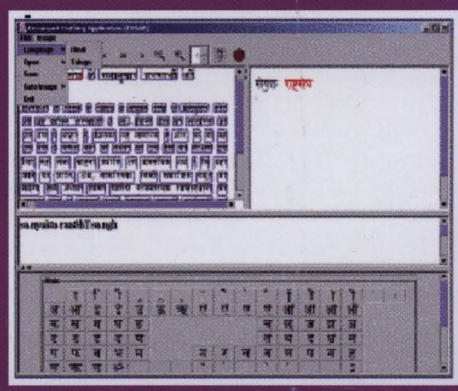
Handwriting



Page Structure



OCR and OCR Correction



Document Analysis Resources



SDIUT '03

