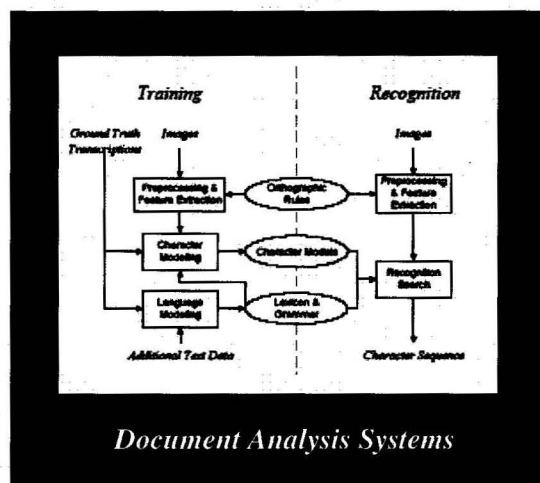
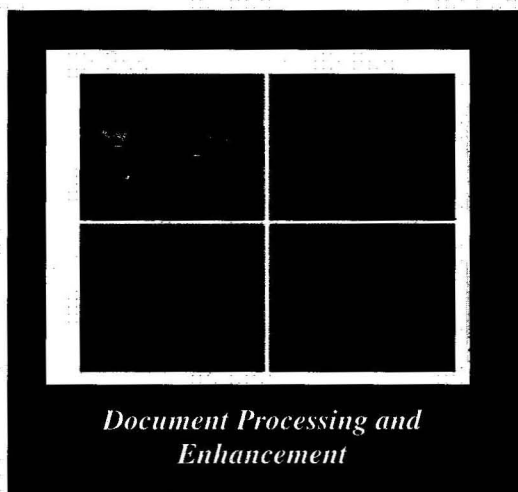


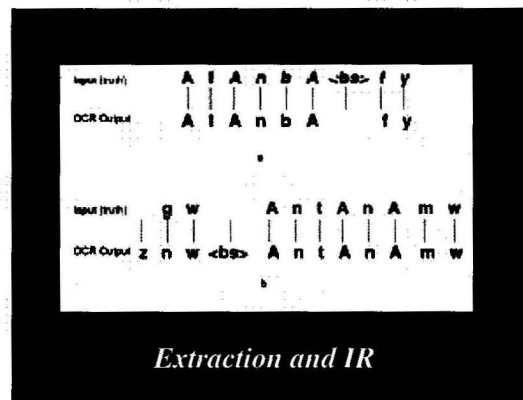
Proceedings

2005 Symposium on Document Image Understanding Technology



Word Shape Taken	Word 1	Word 2	Word 3
0000 0001 0100 0000 0000 0000 0010	مستعمل suitable	مكثف genuinely employed	
0000 0100 0000 0010	مكعب snake cubic	يلعب he plays	يلعب he plays
0000 0000 0000 0000 0001 1000 0100 0100 0000 0101	مستعمل customer of you		
0000 0000 0000 0000 0001 1000 1000 0000 0000	مستورد importer		
0000 0000 0000 0001 0000 0000 0000 0001 0000 0001 0000 0000 1001	مستعملين you will be given a gram	مستمتع you will enjoy yourself	

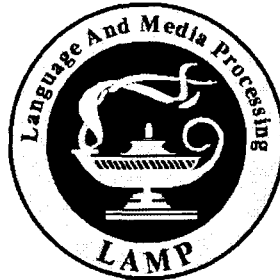
Arabic Document Analysis



November 2-4, 2005
Inn and Conference Center
Marriott Hotel
Adelphi, Maryland

Proceedings
SDIUT05
The 2005 Symposium on
Document Image Understanding Technology

Inn and Conference Center
Marriott Hotel
Adelphi, Maryland
November 2-4, 2005



Sponsored by: The United States Department of Defense
Advanced Research & Development Activity
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: sdiut05@lamp.cfar.umd.edu

Table of Contents

MESSAGE FROM THE ORGANIZERS5

KEYNOTE SPEAKERS

GOOGLE PRINT.....9
Luc Vincent, Google

**EDUCE: ENHANCED DIGITAL UNWRAPPING FOR CONSERVATION AND
EXPLORATION OF INACCESSIBLE TEXTS.....11**
W. Brent Seales and Yun Lin, University of Kentucky

SESSION 1 DOCUMENT ANALYSIS SYSTEMS

**A FLEXIBLE EXPERIMENTATION AND CONFIGURATION PLATFORM FOR
MULTILINGUAL HMM OCR15**
Ehry MacRostie and Premkumar Natarajan, BBN Technologies

DOCUMENT INFORMATION PROCESSING: TOWARDS SOFTWARE AND TEST COLLECTIONS23
G. Agam, S. Argamon, O. Frieder, D. Grossman and D. Lewis,
Illinois Institute of Technology and David D. Lewis Consulting**

DOCLIB: A DOCUMENT PROCESSING RESEARCH TOOL31
Kevin Chen, Stefan Jaeger, Guangyu Zhu and David Doermann,
Booz Allen Hamilton* and University of Maryland*

A PROCESS FLOW FOR REALIZING HIGH ACCURACY FOR OCR TEXT39
Kazem Taghza, Julie Borsack and Tom Nartker, University of Nevada, Las Vegas

SESSION 2 DOCUMENT PROCESSING AND ENHANCEMENT

A DOCUMENT IMAGE ENHANCEMENT MODULE: PERSPECTIVE WARP CORRECTION.....49
C. Monnier, Vitaly Ablavsky, Steve Holden and Magnus Snorrason,
Charles River Analytics, Inc. and Boston University**

DOCUMENT RECOGNITION AND TRANSLATION WITH HANDHELD MOBILE DEVICES.....	55
<i>Esin Darici Haritaoglu and Ismail Haritaoglu, Polar Rain Inc.</i>	

SESSION 3 PROCESSING OF ARABIC DOCUMENTS

A SYSTEM FOR DISCRIMINATING HANDWRITING FROM MACHINE PRINT ON NOISY ARABIC DOCUMENTS	65
<i>John C. Femiani, Mariano Phielipp and Anshuman Razdan, Arizona State University</i>	

NLP-ENHANCED EXPLOITATION OF DEGRADED ARABIC TEXTS	71
<i>Michael Shalev, Encyclopaedia Britannica</i>	

INITIAL RESULTS IN OFFLINE ARABIC HANDWRITING RECOGNITION USING LARGE-SCALE GEOMETRIC FEATURES.....	79
<i>Ilya Zavorin, Eugene Borovikov and Mark Turner, CACI International Inc.</i>	

CLASSIFICATION OF MACHINE PRINT AND HANDWRITING IN MIXED ARABIC DOCUMENTS	89
<i>Karthik Sridharan, Faisal Farooq and Venu Govindaraju, CEDAR, SUNY at Buffalo</i>	

SESSION 4 EXTRACTION AND INFORMATION RETRIEVAL

ACCURATE DOCUMENT CATEGORIZATION OF OCR GENERATED TEXT	97
<i>Robert Price and Anthony Zukas*, Content Analysis LLC and SAIC*</i>	

EFFECT OF DEGRADED INPUT ON STATISTICAL MACHINE TRANSLATION	103
<i>Faisal Farooq and Yaser Al-Onaizan*, CEDAR, SUNY at Buffalo and IBM T.J. Watson Research Center*</i>	

MOBILE INTERACTIVE SUPPORT SYSTEM FOR TIME-CRITICAL DOCUMENT EXPLOITATION	111
<i>George Nagy and Daniel Lopresti*, Rensselaer Polytechnic Institute and Lehigh University*</i>	

SESSION 5 ARABIC DOCUMENT ANALYSIS

HANDWRITTEN ARABIC WORD SPOTTING USING THE CEDAR ARABIC DOCUMENT ANALYSIS SYSTEM	123
<i>Sargur Srihari, Harish Srinivasan, Pavithra Babu and Chetan Bhole, CEDAR, University at Buffalo, SUNY</i>	

WORD SPOTTING IN ARABIC SCRIPT DOCUMENTS.....133
A. Lawrence Spitz and Jim Yaghi, DocRec Ltd.

CHALLENGES IN ADAPTING MACHINE-PRINT ARABIC OCR FOR HANDWRITING.....139
Steven G. Schlosser and Robert C. Vogt, NovoDynamics, Inc.

SESSION 6 CROSS DOMAIN APPLICATIONS

ROBOT NAVIGATION TECHNIQUES FOR ENGINEERING DRAWING ANALYSIS157
Thomas C. Henderson and Chimiao Xu, University of Utah

PERCEPTUAL ORGANIZATION IN SEMANTIC ROLE LABELING.....167
P. Sarkar and E. Saund, Palo Alto Research Center

**PICTOGRAPHIC RECOGNITION TECHNOLOGY APPLIED TO DISTINCTIVE
CHARACTERISTICS OF HANDWRITTEN ARABIC TEXT.....173**
Mark A. Walch and Donald T. Gantz, The Gannon Technologies Group and
George Mason University**

SESSION 7 DOCUMENT ANALYSIS APPLICATIONS

A DOCUMENT TRIAGE SYSTEM FOR ARMY APPLICATION187
F. Fisher, K. Marcus, R. Chang and J. Turner, Army Research Laboratory

MULTI-LANGUAGE HANDWRITING DERIVED BIOMETRIC IDENTIFICATION197
Donald T. Gantz, John J. Miller and Mark A. Walch, George Mason University and
The Gannon Technologies Group**

**TRADEOFF STUDIES ABOUT STORAGE AND RETRIEVAL EFFICIENCY OF BOUNDARY
DATA REPRESENTATIONS FOR LLS, TIGER, AND DLG DATA STRUCTURES.....211**
David Clutter and Peter Bajcsy, University of Illinois

DEMONSTRATION AND POSTER ABSTRACTS

**ABBYY SOFTWARE HOUSE OCR, FORMS PROCESSING AND DATA
CAPTURE TECHNOLOGY REVIEW.....225**
Imelda Valenzuela, ABBYY

VERUS A MIDDLE EASTERN LANGUAGE OCR.....229
Steven G. Schlosser, NovoDynamics, Inc.

**IDG: A BUSINESS INFORMATION EXTRACTION, MANAGEMENT, AND ROUTING
FRONT-END FOR CONTENT MANAGEMENT SYSTEMS231**
*Vikas Krishna, Savitha Srinivasan, Neil Boyette, Isaac Cheng, Jeffrey Kreulen and Tapas
Kanungo, IBM Almaden Research Center*

FAST SKEW AND SLANT CORRECTION FOR ARABIC WRITTEN WORD OR LINE.....235
Essam M. Zaki and Mohamed El-Adawi, Sakhr Software USA Inc.

EVALUATION WORKSHOP ABSTRACTS

PERFORMANCE EVALUATION OF MULTILINGUAL DOCUMENT EXPLOITATION SYSTEMS.....239
Steven G. Schlosser, NovoDynamics, Inc.

ISSUES WITH AUTOMATIC OCR EVALUATION AND METRICS.....241
Kristian J. Concepcion.

EVALUATION ISSUES IN IMAGE REFINER243
Kristen Summers and Eugene Borovikov, CACI

THE SPORADIC NATURE OF OCR EVALUATIONS245
Tapas Kanungo, IBM Almaden Research Center

**GROUND TRUTH REPRESENTATION USED IN TESTING AND OPTIMIZATION OF THE
OPTICAL WORD RECOGNITION SYSTEM.....247**
Mike Ladwig, Northrop Grumman Corporation

METRICS FOR WORD SPOTTING249
Sargur Srihari, CEDAR University of Buffalo

EVALUATING WITH INFORMATIONAL CONFIDENCE251
Stefan Jaeger and David Doermann, University of Maryland

AUTHOR INDEX.....253

Message from the Organizers

Over the past 10 years, SDIUT has brought together thousands of researchers from academia, industry and government to continue to explore the unique needs and challenges facing the government with respect to processing a tremendous volume on hard copy material they encounter every day. With continued effort, and increased awareness of the need for such capabilities, we will be able to turn the ideas being discussed in this meeting, into meaningful solutions for the future.

This year, the Symposium will host over 20 technical talks, almost a dozen posters and demos and 2 keynote talks, followed by a one day workshop on the Evaluation of Document Image Processing Technologies, run by Ms. Jen Doyon of the Mitre Corporation. The workshop will bring together and focus experts in the field of document image processing on evaluation issues with the hope to making progress toward enhanced ground truth representation and corresponding evaluation techniques. Expanding on the character and word accuracy based metrics in current use facilitates a more complete approach to evaluating the performance of established recognition systems. It also enables component-level evaluation of specialized technologies such as script / language identification, line and word finding, signature detection, handwriting recognition, word spotting, forms processing and others. We hope that this session will help define solutions that can be incorporated into future government programs.

The organization of this Symposium would not have been possible without the endless hours of work by the staff of the University of Maryland Institute for Advanced Computer Studies (UMIACS) and in particular of Ms. Denise Best. Thank you Denise for your wonderful work.

Thank you once again for your participation and contributions to this Symposium.

The 2005 SDIUT Organizing Committee.
November 2, 2005

Keynote Speakers

GOOGLE Print

**Luc Vincent
Google, Inc.**

EDUCE: Enhanced Digital Unwrapping for Conservation and Exploration of Inaccessible Texts

W. Brent Seales Yun Lin
Computer Science Department
University of Kentucky
{seales, ylin}@netlab.uky.edu

This work addresses digital unwrapping of a class of damaged and fragile objects that are impenetrable for conventional imaging equipment and cannot easily be opened physically for a clear digitization. Existing document processing/analysis techniques are not able to reveal a considerable amount of the contents wrapped inside. We develop a general approach for building a readable image of an opaque, rolled or folded document from a non-destructive volumetric scan. The problem is framed by localizing, constructing and manipulating a texture image induced by a surface embedded in a 3D voxel space. Our approach is characterized as an iterative energy-based optimization framework. The goal is to form an optimal texture image by optimizing the surface geometry using a $2\frac{1}{2}$ Active Contour model and a novel texture-guided contouring algorithm. As one of the critical steps in the processing framework, a physically-based surface parameterization algorithm is used to map the convoluted embedded surface to an image plane which preserves angles, lengths and minimizes the distortion of text in the image. Initial experimental results on sample scrolls show the feasibility of this method, which we believe is necessary for scholars seeking to study inaccessible texts. Our vision is that these methods will lead to viable techniques for scanning everyday opaque objects such as books without opening them.

Computed Tomography (CT), a widely used technique in healthcare, is nondestructive, penetrating and revealing, which makes it a good candidate to use in folded document digitization. Our processing framework takes volumetric data from a custom CT scanner (with tunable X-ray energy and scan resolution) as input. The output data of a scan is treated as a cuboid of aligned voxels in x,y,z directions. Ink, document substrate and empty space can be distinguished from the intensity variations in the unstructured voxel set.

We build an optimization algorithm to correctly localize the substrate surface within the voxel set.

The algorithm enforces two levels of optimization constraints. One is imposed by a $2\frac{1}{2}$ Active Contour segmentation model and the other, which we call texture-guided contouring, is based on texture image analysis which takes advantage of the surface parameterization that maps the textured surface mesh to an image plane. The steps of the algorithm proceed as follows:

1. Construct an optimal mapping M_1 from a volume V to a piecewise smooth surface S :

$$M_1 : V \rightarrow S$$

by minimizing the energy E_1 , where E_1 is defined as:

$$E_1(V, S) = I(V, S) + C(S)$$

E_1 is a function of the volume V and an embedded surface S . $I(V, S)$ represents an Image component which evaluates how well the surface locates the intensity features contained in the volume. $C(S)$ represents a Shape component which evaluates the smoothness of the surface.

2. A filter is applied on V and S that produces a *nonregularized* texture map T_1 :

$$F : V + S \rightarrow T_1$$

S is represented as a triangulated mesh and its triangular components map to R^2 independently, where the texture map T_1 is defined without inheriting any adjacency relationships.

3. A mapping M_2 is constructed from the surface S representing the document to a planar surface U , which converts the nonregularized texture map T_1 to a regularized texture map T_2 :

$$M_2 : S + T_1 \rightarrow U + T_2$$

T_2 is expected to represent the “image” of the information carried on the surface of the document. We approximate a conformal mapping by using mass-spring system simulation.

4. Further optimization is enforced by minimizing the energy E_2 for T_2 , which can be framed as:

$$E_2(T_2) = P(T_2) + R(T_2)$$

P is a generalized function evaluating low level image features while R is a generalized function based on high level document image understanding techniques. The surface is optimized iteratively through steps 2,3,4. During each iteration the surface position is updated along its normal direction.

5. Return U, T_2

We build a $2\frac{1}{2}$ Dimensional Active Contour model to construct M_1 in step 1. We take advantage of cross-sections to define an initial surface by approximating 2D salient contours in representative slices. Optimization follows and in this phase optimization is only performed based on raw volume data and the surface, ignoring the texture constraint for the moment. An optimal mapping is defined as minimizing the energy function E_1 , which is a linear combination of an Image component $I(V, S)$ and a Shape component $C(S)$. $I(V, S)$ calculates the gradient of the surface with respect to the volume, which attracts the surface to contours with large intensity gradients. $C(S)$ is evaluated as a combination of 5 components, which are first-order and second-order partial derivatives of the parametric surface S . These components correspond to continuity, curvature and twist. The surface S is approximated by a mesh decided by a finite set of points. We solve the optimization problem using a greedy algorithm, which is an extension of planar active contouring to $2\frac{1}{2}$ dimensions: points move on the discrete grid to any position within a defined planar neighborhood while the constrained energy functional is built in three dimensions.

The mapping M_2 that unwraps the embedded surface plays a significant role in the quality of the resulting texture. The goal is to introduce as little distortion as possible to the text that may be present in the document. The point is that the surface representing a document can have *arbitrary shape*. We approximate a conformal mapping by using mass-spring system simulation. A surface mesh, modeled as a mass-spring system, performs realistically when simulation principles are founded on physical laws. An optimal mapping is achieved when the system reaches equilibrium status with minimal potential energy. The mass-spring system strives to maintain the length of each edge while performing mapping by deformation. For a triangle, when its three sides are preserved, its three angles are also preserved. Since the spring system minimizes edge lengths globally,

i.e., the length summation of all edges, it approximates but does not produce a strict conformal mapping.

This surface parameterization method has an important advantage over other methods: arbitrary surfaces can be handled. For surfaces that are rolled, folded, or wrapped, such as a scroll, we use constrained dynamics to perform an unrolling operation followed by flattening. The virtual unrolling “holds” one edge of the surface and allows the free portion to descend under simulated gravity.

We assume a correctly positioned surface will induce a texture map T_2 that contains both high and low frequency components. Further optimization of surface location is achieved by adding a novel texture constraint in step 4 that ultimately results in optimization directly over the texture. We adopt a sharpness measure for the texture image based on frequency domain analysis and entropy evaluation. After each loop, the surface S is updated along the normal in a limited neighborhood, yielding a new surface S' . A texture map T'_1 is generated for S' in step 2 and a regularized texture T'_2 by unwrapping and flattening S' in step 3. If its texture evaluation energy is reduced, the update S' is accepted. General low-level analysis is a starting point. Our framework is extensible by including the possibility to apply advanced pattern analysis based on a priori knowledge of the document’s historical and cultural milieu.

We have implemented the key algorithms outlined above and have conducted a series of tests on replicas and experimental artifacts scanned using custom CT scanners. In the first testing phase, we scanned and processed regularly shaped canvas scrolls and irregular papyrus scrolls. Motivated by that success, we created a set of replicas from papyrus and ink and embedded them in polyurethane plastic. Two representative examples include a papyrus strip rolled and a papyrus strip connected into a Mobius band with ink (locally) on both sides. The results of applying our approach successfully reveals the material content without physically opening them. These successes demonstrate the potential application of our algorithms to more difficult but important cases such as ancient scrolls and damaged books. The post-processing algorithms presented here are a necessary though not sufficient step toward the goal of reading texts, books, and documents from volumetric scans. We anticipate improvements in scanning technologies in the areas of resolution and sensitivity to materials, which together with work such as ours will move forward the analysis of documents from penetrating scans.

Document Analysis Systems

A Flexible Experimentation and Configuration Platform for Multilingual HMM OCR

Ehry MacRostie Premkumar Natarajan
{emacrost,pnataraj}@bbn.com
BBN Technologies
Cambridge, MA

Abstract

In this paper we present the BBN Flex multilingual OCR model configuration and experimentation platform which allows non-expert users to configure and test HMM models for the BBN Byblos OCR system. The Flex platform is designed to provide customers with model training capabilities for any machine printed language. Furthermore, training capabilities can be updated as new modeling techniques become available. This paper describes the HMM OCR modeling paradigm and the design of the Flex platform. Modeling techniques currently available on the platform as well as new techniques that we plan to add are discussed. Finally, we describe a test of the system on Korean machine printed data in which we achieved a character error rate of 6.84%.

1 Introduction

One of the major challenges for Optical Character Recognition (OCR) systems is coping with the variety of data encountered in real-world settings. Document images may be printed by a number of different physical means in any number of different fonts. Digitization of the document by means of a scanner, fax or camera may introduce distortion and noise to the document. Binarization of color or grayscale digital images can further degrade image quality. Furthermore, all of these problems may be encountered in documents in languages for which no OCR system exists. In government settings such real-world problems are often more pronounced.

The BBN Byblos OCR system is a data-driven Hidden Markov Model (HMM) system that can be configured for a wide variety of languages and document degradation types [1,4,5]. The goal of the recent effort described in this paper is to simplify the model configuration process so that customers can easily configure the Byblos system for their data and language requirements. In addition, we wanted to create a system that allows us to easily deploy new model training techniques developed at BBN that increase the performance and speed of our OCR system.

This paper is organized as follows. In Section 2 we describe our basic HMM modeling paradigm for OCR. In Section 3, the Flex platform is described and some details regarding our use of the Unicode character encoding standard are discussed. Section 4 addresses HMM modeling techniques supported in the current version of this platform. In Section 5 we discuss some modeling techniques that we plan to incorporate into the platform in the future. In Section 6 we describe the results of testing the system on a corpus of Korean documents.

2 The HMM OCR Modeling Paradigm

In this section we briefly review the BBN Byblos OCR system. For a more detailed description the reader is referred to [1]. A pictorial representation of the system is given in Figure 1. In the figure, the ellipses represent script-dependent knowledge sources. The conceptual script-independent system components are identified by rectangular boxes.

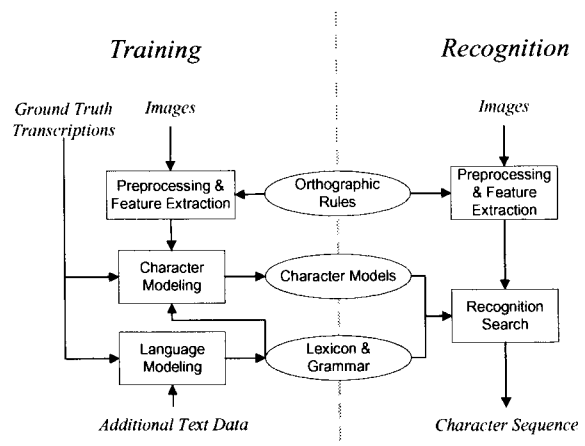


Figure 1: HMM OCR overview

The OCR system can be divided into two basic functional components: training and recognition. Both training and recognition share a common pre-processing and feature extraction stage in which we

first de-skew the scanned image and then locate the regions (bounding boxes) of individual text lines. We then compute a feature vector as a function of the horizontal position within each of these line regions. Each line of text is horizontally segmented into a sequence of thin, overlapping, vertical strips called frames. For each frame we compute a language-independent, feature vector that is a numerical representation of the frame.

The OCR system models each character with a multi-state, left-to-right HMM. As described in Section 3.3, the system handles languages with right-to-left, bi-directional and other rendering orientations by normalizing them to a series of left-to-right feature vectors internally.



Figure 2: Sliding window feature extraction

Each state of the HMM has an associated output probability distribution over the features. The number of states and the allowable transitions are system parameters. For our full-character experiments we use 14-state HMMs with the topology shown in Figure 3. While the number of states can be changed, the basic topology containing self-loops and state skips is the same for all configurations and all languages.

Training is performed using the Baum-Welch or Forward-Backward algorithm that aligns the feature vectors with the character-models to obtain maximum likelihood estimates of HMM parameters. The state output probabilities of the HMM are modeled as mixtures of Gaussians. The HMM parameters estimated during training are the means and variances of the Gaussians, the mixture component weights, and the state transition probabilities. 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 is estimated from a suitably large text corpus. Typically the grammar (language model), which provides the probability of any character or word sequence, is also estimated from the same corpus.

A significant advantage of HMM-based systems is that they provide a language-independent framework for training and recognition. At the same time, they do not require the training data to be segmented into words or characters – they automatically train themselves on un-segmented data.



Figure 3: Left-to-right HMM topology

This data-driven approach allows custom models to be trained for new languages or domains with minimal data preparation and transcription costs. The Flex platform allows non-expert users to perform training on-site with their own data.

3 The Flex Platform

3.1 Overview of the Flex Platform

The Flex platform consists of three main components, the Byblos model training toolkit, the Condor High Throughput Computing system (from the University of Wisconsin), and a Microsoft ASP.NET Web User Interface (UI) layer. The entire system runs on a single Windows 2000 Server machine providing access from many computers via its web interface.

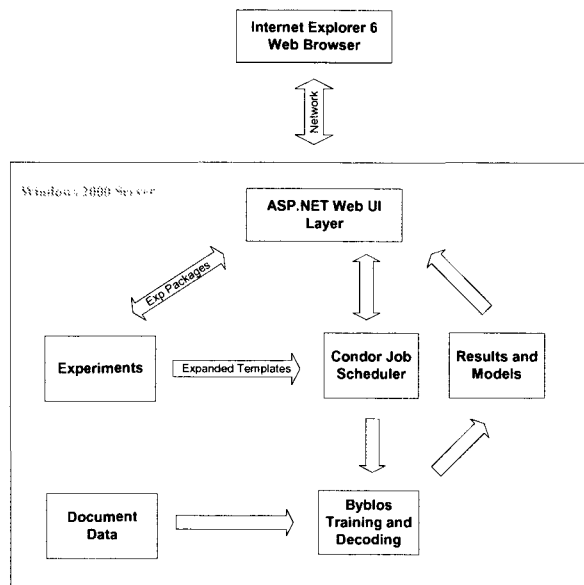


Figure 4: Platform components

The Condor and ASP.NET components are standard off the shelf software packages that provide the parallel processing and application layer support. Condor provides the parallel processing environment for the system. Though the current version of the Flex platform runs on a single machine, it can easily be configured to include a number of parallel compute nodes. The application logic is implemented in an ASP.NET layer. This layer manages experiments, exposes selected Byblos parameter values to the user, and displays recognition results. The Byblos OCR training suite is a collection of programs developed at BBN that includes

all of the HMM OCR model training and recognition functionality depicted by the rectangular boxes in Figure 1.

The platform infrastructure is built around logical units called experiments. An experiment encapsulates one type of high-level process and typically contains a number of sequential and/or parallel steps. In their generic form, all experiments consist of a central script that controls the sequence of programs to be executed, an XML parameter file that links the user interface to the experiment, and a number of generic parameter files that contain runtime parameters for the programs. The user interface is designed so that any number of experiments can be dropped into the system without any reconfiguration. Many common tasks such as splitting a corpus into test and training sets and converting transcriptions into a format that is usable by Byblos have been implemented within the experiments. The goal is to not require the user to perform any data manipulation other than loading the data onto the server.

3.2 User Interface

The system user interface is designed to be as simple as possible while at the same time enabling the user to easily see what is happening in the training process. We have implemented several new visualization tools that allow the user to easily view the status and progress of the training process.

For each new data set, the user first selects the appropriate type of training. This selection requires some knowledge of the training techniques available on the platform. A selected experiment type is then configured using a standard HTML form.

The parameters available using the user interface vary depending on the experiment type. Some common parameters include the data corpus location, the language being processed, and the image resolution of the document images being processed. Character modeling experiments allow the user to select whether or not to train a new Linear Discriminant Analysis (LDA) matrix and whether the reference transcriptions should be normalized using the Unicode NFKC normalization type. For the sub-character experiment, the user must provide the total number of sub-characters to use for training as described in Section 4.4.

Once the experiment is configured, it is then submitted for processing by the Condor system. As jobs run on the queue, their progress can be monitored using a job monitor tool.

One of the new visualization tools developed for this platform was the experiment explorer. The experiment explorer, shown in Figure 5, displays the parameter files and log files for each step in the experiment. It provides a graphical view of the experiment structure as well as a convenient way to monitor and debug an

experiment.

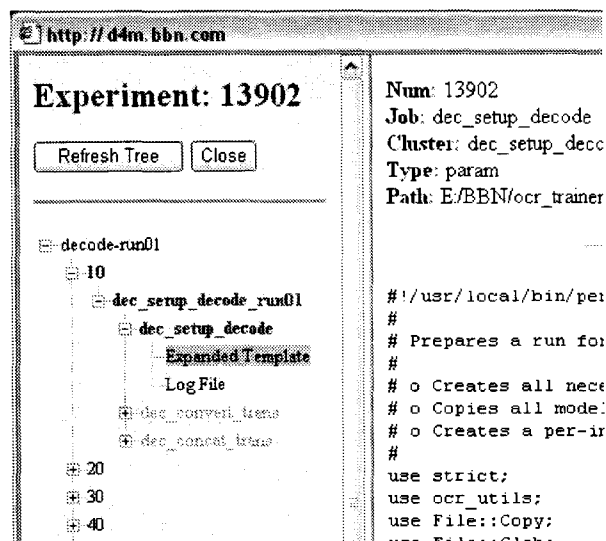


Figure 5: Experiment explorer display

After training, the models are evaluated against a test set. The test results are displayed in a results viewer tool shown in Figure 6. In the results viewer, the original line image, the corresponding recognition result, and the reference transcription are displayed together. The result and reference transcriptions are aligned and color coded so that errors can easily be seen. Line-by-line, page-by-page, and overall error rates are also displayed.

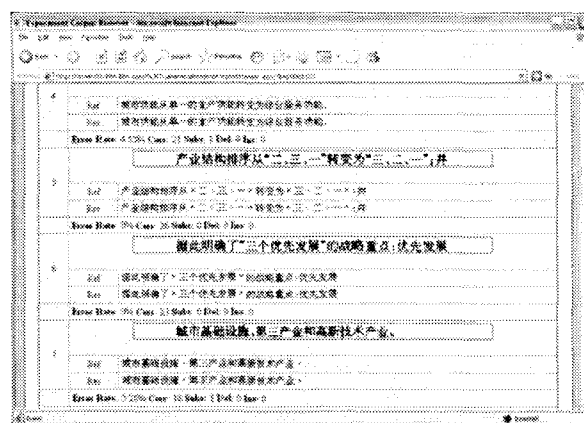


Figure 6: OCR recognition results display

3.3 Data Requirements

In the Flex platform we assume that the user will provide a corpus of document images and the corresponding line-by-line transcriptions.

A critical benefit of our HMM OCR technology is that it does not require manual character-level segmentation of document images for training. Also, line locations are found automatically in black and white input images using our HMM line finding method

[1]. This greatly minimizes the effort required to transcribe a training corpus.

We have standardized our OCR system on the Unicode character encoding standard in order to support a wide variety of languages and to facilitate easy integration with other downstream processes. Even within the Unicode standard there are a number of character-rendering issues that our system must deal with.

One of the most common issues is the reading direction of text. While most Western languages are read from left to right, Arabic and some other Middle Eastern languages are read right to left. Our system accounts for this by reversing the direction of feature extraction to match the reading direction and training models in the usual way. This requires a priori knowledge of the language of each document.

Reading direction becomes a little more complex when bi-directional text rendering [8] is taken into account. Unicode is always encoded electronically in speaking order. When a line of text contains right to left text interspersed with left to right strings, the left to right strings are rendered as such within an overall right to left line.

For our modeling purposes in order to align the feature vectors with their corresponding Unicode transcriptions, we reverse the underlying Unicode speaking order for embedded bi-directional strings temporarily for modeling and recognition then map them back to the standard Unicode ordering in the recognition output.

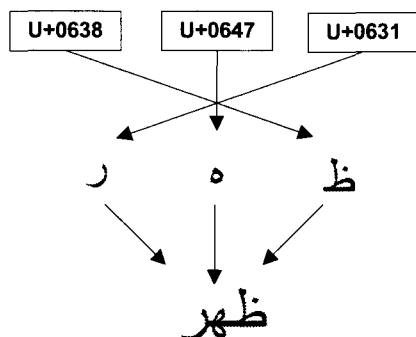


Figure 7: Right-to-left rendering in Arabic

There are a handful of characters that are rendered out of sequence with the general rendering direction of the script from which they come. An example of this is the Devanagari vowel sign 'I' (encoded as U+093F in Unicode) used in Hindi and other Indic languages. When it appears to the right of a consonant in the character encoding, the 'I' is rendered to the left of the consonant even though standard rendering for Devanagari is left to right. An example of out-of-sequence character rendering is shown in Figure 8.

A related, though more frequent character rendering issue occurs in languages such as Thai and Hindi. These languages have vowels and diacritics that are rendered

in the same horizontal space as surrounding consonants. Since our feature extraction procedure proceeds in one direction using a fixed-height window, features from multiple characters are folded into the same feature vectors when two characters are rendered in the same horizontal space.

In order to deal with cases of overlapping and out-of-sequence character rendering, we take a combinatorial approach. Instead of detecting vertical or out-of-sequence rendering during decoding, we detect these cases in the training transcriptions and create unique models for each one. In practice this is done by clustering adjacent characters in the training transcriptions based on rendering information. The clusters are then used as the base modeling units.

Using such clustered modeling units, on a corpus of clean scanned Hindi documents, we have achieved character error rates of 1.4% using the full-character model configuration [5].

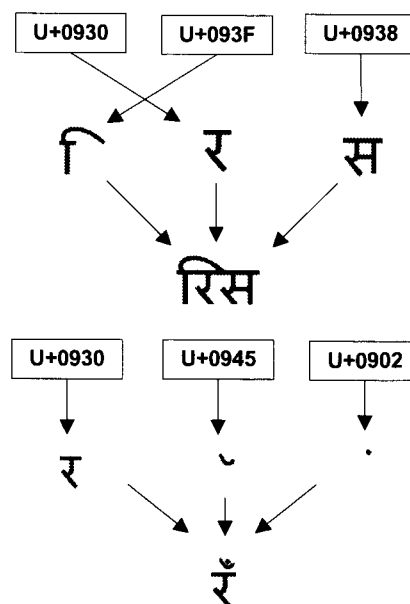


Figure 8: Vertical rendering and out-of-sequence rendering in the Devanagari script

4 Supported Modeling Techniques

This section describes modeling techniques that are currently supported by the Flex platform. Full-character modeling is the base modeling technique that usually provides the highest decoding accuracy. Several other modeling configurations that increase decoding speed over the full-character configuration are described; these include Fast Gaussian Trees, Tied Mixture Models and Sub-character models.

4.1 Full-character Modeling

Character Tied Mixture (CTM) training is the basic full-character training method used by the OCR training system. It is the preferred training method for most alphabetic scripts. For scripts with large numbers of

unique characters, sub-character modeling offers significantly faster decoding speeds.

The CTM training method trains a separate HMM for each unique character in the training corpus. Typically we use 14-state, left-to-right HMMs with self-loops and one-state skips. A codebook of 256 Gaussians is associated with the HMM for each character and there is no sharing of Gaussians across different characters.

CTM models usually produce the most accurate decoding results and the slowest decoding speed. Apart from the model parameters, decoding speed for CTM models depends on the total number of unique glyphs that we choose to model for a particular language. These glyphs are usually simply the characters in the target language, but in the case of languages with vertical character rendering some glyphs are combinations of characters.

4.2 Fast Gaussian Trees

Broadly speaking, the time spent during recognition is spent either in computing the HMM score (i.e., evaluating the Gaussian likelihoods) or in executing the search logic itself. In a basic HMM system, typically the Gaussian computations tend to dominate the overall computation time. One efficient method for reducing the Gaussian computation time is to first select (using a fast computationally cheap procedure) a preferred list of Gaussians for each feature vector and then only compute the likelihoods for those preferred Gaussians. Our implementation of this technique is referred to as Fast Gaussian Computation (FGC) trees or just Fast Gaussian Trees. For each feature vector, we walk down the FGC tree to find the preferred list of Gaussians for that vector.

FGC trees are automatically estimated from training data using trained CTM models. They do not alter the base CTM models and can be turned off during recognition.

While the decoding speed is faster than straight CTM models, recognition accuracy is typically a little worse using FGC trees. One important feature of the FGC tree architecture is that we can trade-off accuracy for speed by adjusting the length of the list of preferred Gaussians; the shorter the list, the greater the gain in speed and (typically) the greater the loss in accuracy.

4.3 Tied Mixture Modeling

Another way to reduce the time spent in Gaussian computations is to share Gaussians across different characters. We call this sharing approach the Tied Mixture (TM) model. In the TM model, instead of using a separate set of Gaussians for each character model, a single set of Gaussians is shared among all models. Mixture weights for each model are used to distinguish models from one another. As a result, only one set of Gaussian likelihoods needs to be evaluated

for each input feature vector. In the standard CTM decoding, we use a separate set of 256 Gaussians for each character HMM, whereas in the TM mode we typically use a single set of 512 Gaussians. This is a huge reduction in the number of Gaussians in the system. As is to be expected, this reduction in the total number of parameters leads to a worse performance but once again the trade off between accuracy and speed can be controlled by adjusting the number of Gaussians in the TM model.

Our recognition engine uses a patented two-pass beam search. The goal of the first pass is not to generate the single-best hypothesis; rather, the goal is to prune out as many unlikely hypotheses as we can without rejecting the correct answer. The pruned hypothesis list is the starting point for the second pass where more detailed models (e.g., CTM models) can be used. We have found that the TM models are most effective method when used in the first pass in combination with CTM models in the second pass.

As with FGC trees, combining TM and CTM models increases decoding speed at a small cost to decoding accuracy. FGC trees and TM models can be used jointly to obtain a greater increase in speed than either approach can deliver when used separately.

4.4 Sub-Character Modeling

Unlike the FGC trees and TM models, the primary aim of the sub-character modeling approach is to reduce the time spent in the search logic. For large character sets, the fan-out at each node in the search network is also large. As a result, for languages with large character sets, the time spent in the search procedure (i.e., propagating scores and hypotheses from a node to all following nodes and sorting node scores at each time instant), can become a large fraction of the total compute time – especially when FGC trees and TM models are used to reduce the Gaussian computation time.

The sub-character configuration is based on the observation that when looking at a group of characters closely, many sub-regions are similar or identical across different characters. For example the first half of the character ‘o’ and the first half of the character ‘c’, look virtually the same, and therefore should not require different models. This similarity is exploited by modeling similar sub-characters with a single model then chaining these models together to model the full characters. Typically, each sub-character model is a 3-state HMM and a full character is spelled using up to five sub-characters to create a 15-state model.

In keeping with our overall approach of not requiring language experts, the list of sub-characters and the spelling of each full character using these sub-characters are both performed automatically using a clustering procedure. The optimal number of sub-characters could vary depending on the language. We

have used sub-character models for Chinese and Japanese data sets containing approximately 4000 and 3000 full characters, respectively. For these data sets, we have found that 1024 sub-characters is the best configuration [4].

5 Planned Improvements

Duration modeling and Adaptation are two modeling techniques that have been developed at BBN but have not yet been integrated into the Flex platform. Duration Modeling increases decoding speed (without any loss in accuracy) by adding a temporal constraint for each character. Adaptation allows existing models to be tuned for a new domain by using a small amount of training data from that domain.

5.1 Duration Modeling

The speed improvements described above involved reducing the number of Gaussians computed during character recognition. Another approach that has been explored is to use a constraint on the length of a character to speed up the forward pass of decoding. This is known as Duration Modeling [6]. For the purpose of duration modeling, the duration (or length) of a character hypothesis is simply the number of frames that are aligned with that hypothesis. Please refer to Section 2 for a description of frames.

As described earlier, the Byblos recognition engine uses a two-pass search algorithm. In the fast match pass, unlikely hypotheses are pruned from the search space of hypotheses. The goal here is to remove as many wrong hypotheses as possible before the detailed match pass. Duration Modeling adds a constraint on the length of a character during the fast pass match to prevent new hypotheses from being generated at improbable locations. For example, the duration model would prevent a character hypothesis that has one-tenth the duration observed in training.

Character durations are modeled as simple histograms on the durations observed during training. We use a separate histogram for each character. The trained histogram is then smoothed using an appropriate smoothing window, and the smoothed histogram is used to apply a hard constraint in the search algorithm.

Experimental results on Chinese data have shown that Duration Modeling can improve decoding speed by 37% without decreasing decoding accuracy [6].

5.2 Adaptation

Another potential addition to the Flex platform is character model adaptation. Often models exist for a particular language of interest; however the characteristics of the target domain of documents to be processed do not match the characteristics of the training corpus that was used to train the existing models. This mismatch often results in degraded performance. For example, the models may be trained

on data from newspapers and magazines while the domain of interest might be photocopied, faxed, or camera-captured documents.

Adaptation is a method that allows us to use a small amount of new training data from the domain to tune the parameters of existing trained models to that domain. Previous work at BBN [7] has shown that Maximum Likelihood Linear Regression (MLLR) adaptation can provide significant improvements in performance.

6 A Case Study: Training Korean Models

During development, the Flex platform was continually regression tested using a number of data corpora that we have used in the past for model training. These corpora included data sets from Hindi, Arabic, Thai, Chinese and Japanese. Once the initial version of the platform was developed, we wanted to test the system on a completely new corpus/language to verify that our end-to-end training procedure could handle a new language.

We tested using a corpus of 1054 clean Korean images generated electronically from text. Before this test, we had never trained our system on any Korean data.

The corpus was not manually inspected ahead of time. The black and white documents and the corresponding UTF-8 transcriptions were simply loaded into the system and the standard procedure was used to validate the data and to run the model training process. The system automatically split the corpus into training and test sets – 811 documents in training and 243 in test. The ratio of training to test is a configurable parameter available to the user so these amounts could have been different. A total of 1525 units were modeled including individual Unicode code points and character clusters as described in Section 3.3.

The first experiment we ran was full-character training. This took approximately 18 hours of processing time on our dual-processor 2.6 GHz machine but only about 30 minutes of setup time for the user. The experiment ran through to completion without error. We ran recognition followed by optimization of the decoding weights then decoded again using the new weights. The character error rate on the fair test set using the CTM models was 6.84%.

Experiment	Character Error Rate
Full-character (CTM)	6.84%
CTM plus TM &FGC	7.41%
Sub-character	7.52%

Table 1: Error rates for clean Korean images

The standard procedures were followed to train FGC trees, TM models, and CTM models. Additionally, we used the standard procedure to train sub-character

models. The recognition results for all of these experiments are shown in Table 1.

7 Conclusions

The Flex OCR model configuration platform greatly simplifies the creation of new models for the BBN Byblos OCR system. It will allow non-expert users to train OCR models in their own facilities on their data. The platform includes a number of new tools for visualizing various aspects of the model training process. Updated modeling techniques that provide better performance and recognition speed can easily be added to the platform as they become available.

The BBN Byblos OCR system is now standardized on the Unicode character encoding system which allows the system to support a wide variety of languages.

Though this platform was developed for OCR modeling we believe that the distributed computing environment and the user interface can be easily re-configured to support other similar tasks.

Acknowledgements

This work was funded by the Army Research Laboratory through a LASER ACTD effort. We would also like to acknowledge the support and encouragement of Mr. Luis Hernandez of ARL.

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, pp. 43–63.
- [2] Z. Lu, R. Schwartz, P. Natarajan, I. Bazzi, and J. Makhoul, "Advances in the BBN BYBLOS OCR System," *Proc. of Intl. Conf. Doc. Analysis and Recognition*, Bangalore, India, 1999, pp.337–340.
- [3] P. Natarajan, R. Schwartz, M. Decerbo, T. Keller, "Porting the BBN BYBLOS OCR System to New Languages," *Symposium on Document Image Understanding Technologies*, 2003, pp. 47–52.
- [4] E. Macrostie, P. Natarajan, M. Decerbo, and R. Prasad, "The BBN Byblos Japanese OCR System," *International Conference on Pattern Recognition*, Cambridge, UK, Aug. 23-24, 2004.
- [5] P. Natarajan, E. MacRostie, M. Decerbo "The BBN Byblos Hindi OCR System," *Document Recognition and Retrieval XII, Proceedings of SPIE Vol. 5676*, 2005, pp 10-16.
- [6] P. Natarajan, R. Sundaram, R. Prasad, E. MacRostie, "Character Duration Modeling for Speed Improvements in the BBN Byblos OCR System," *International Conference on Document Analysis and Recognition*, Seoul, Korea, 2005, pp. 1136-1140.
- [7] P. Natarajan, et. al., "Robust OCR of Degraded Documents" *International Conference on Document Analysis and Recognition*, Bangalore, India, 1999, pp. 357-361.
- [8] Davis, M. "The Bidirectional Algorithm," *Unicode Standard Annex #9, Revision 15*, March 2005.

Document Information Processing: Towards Software and Test Collections

G. Agam, S. Argamon, O. Frieder, D. Grossman
Illinois Institute of Technology
Department of Computer Science
10 W. 31st Street, Chicago, IL 60616

D. Lewis
David D. Lewis Consulting
858 W. Armitage Ave., #296
Chicago, IL 60614

Abstract—Analysis of large collections of complex documents is increasingly critical to forming accurate and precise intelligence appraisals of enemy activity. Complex documents include handwritten notes, diagrams, and graphics in addition to printed and formatted text. Collections may comprise many different types of complex documents and data, hence extracting useful intelligence from such collections is difficult and time-consuming. Existing automated analysis and search tools are of minimal help. Point solutions exist for some relevant subproblems, but most of these systems are specialized to work with only a single type of information, and in some cases are specific to a particular genre.

We describe issues surrounding our ongoing development of an *integrated* system supporting complex document information processing (CDIP). When completed, the system will support: (a) segmenting and analyzing complex document components to produce data representations supporting querying and data mining, (b) importing such analyzed documents into database structures that allow efficient querying and data mining, (c) answering complex queries against information spanning multiple kinds of complex document data, and (d) discovery of patterns and relationships in complex collections, optionally guided by user queries. We also discuss our initial efforts to build a complex document test collection for evaluating both our system and others that may appear in the future.

I. INTRODUCTION

In the modern geopolitical climate, very large collections of complex documents are increasingly important to form accurate and precise intelligence appraisals of enemy activity. Such collections are rife, including many documents recently captured in Iraq and Afghanistan by U.S. forces, those captured in Ramallah by Israeli forces, and more. Effectively extracting useful intelligence from such large collections is currently an incredibly difficult and time-consuming task, since traditional automated approaches are not applicable. Analysts currently dealing with complex document collections must manually sift through enormous and jumbled collections of typed and handwritten texts, diagrams, lists, and more for useful facts. For instance, in the hunt for Saddam Hussein a

team of military officers laboriously scoured thousands of complex documents by hand and gradually built a complicated multicolored chart showing links between people and organizations, to make sense of the enormous mass of information confronting them. As no integrated automated tools currently address this problem, there is a growing and clear need for computer systems for *complex document information processing* (CDIP). This paper describes our efforts towards development of a useful CDIP system.

We consider *complex documents* to include handwritten notes, diagrams, and graphics in addition to printed and formatted text. Furthermore, collections can consist of multiple types of documents; as a simple example, the Enron collection contains e-mails, policy documents, business statements, and more. Currently, no extant systems focus on complex documents; rather various point solutions exist for subproblems. Solutions exist for OCR of document images – solutions exist for recognition of entities in text – solutions exist for handwriting recognition – but *no* solutions exist that integrate all needed methods for processing complex documents.

The essential problem is that *existing point solutions take no regard for the entire complex document problem*. Research progress is also slowed by lack of any accepted complex document test collection. This paper describes our work on developing an architecture that supports complex document information processing, building a complex document collection for evaluation, testing our architecture using core (existing) point solutions, and improving overall performance by identifying bottlenecks and developing new point solutions or integration techniques, as needed.

A. Background

Though no integrated CDIP system exists, to the best of our knowledge, point solutions do exist for all key components of the complex document problem. Some are addressed by NIST-funded data collection and com-

petitive evaluation efforts such as TREC, CLEF, NTCIR, and MUC. They include:

- optical character recognition (OCR)
- extracting document structure
- handwriting and signature recognition
- extraction of tables/lists and their data
- finding/interpreting figure captions
- search/retrieval of images and graphics
- cross language information retrieval
- entity and structured information extraction
- document clustering

However, such systems are specialized—each only works with a single type of information, and is often specific to a particular genre (such as news stories). Thus, when faced with complex documents, users must partition the data manually and use multiple systems in a piecemeal fashion, manually collating results. Three main problems arise when using such systems piecemeal for CDIP.

First, when a specific system is unaware of the larger context of the information which it is given, its results may be degraded, as well as incomplete. For example, if a printed-text OCR system is given a printed page with overwritten handwriting, it will likely give a high error-rate on the printed text, as well as possibly producing extra, spurious, recognition results for the handwritten portion. An integrated system that first separates printed text from handwriting and processes them separately would do much better.

Second, when multiple information processing systems are used, where each system has no awareness of the others, the difficult tasks of collating, cross-checking and determining consistency of information items must be done by hand. This is both expensive and error-prone when dealing with very large numbers of documents.

Third, component systems normally require the determination of operation parameters which are set to optimize performance. Optimizing the performance of individual components, however, does not guarantee optimal performance of the overall system.

B. Project goals

Today's fragmented solutions offer text, structured data, and XML search, but lack a unified search regardless of the inherent structure of the data. Today, analysts are forced to search for key pieces of evidence by hand, log them into a journal, and then gradually identify connections among information items. We are currently developing an integrated suite of software tools targeted at *complex document information processing* (CDIP). These tools will ultimately support:

- Posing of complex queries against information spanning structured, semi-structured and unstructured data in a complex document collection,
- Analysis of each component of a complex document collection with appropriate techniques (e.g., document analysis for scanned text, language processing for online text image analysis and OCR output, statistical transformations for numerical data) to produce attributes that support querying and data mining,
- Importing of analyzed documents into database structures that allow efficient querying and data mining, and
- Discovery of patterns and relationships in complex collections, optionally guided by user queries.

All tools are being implemented using a modular design supporting a variety of applications; our goal is ensuring that such tools are suitable for use with existing analytical tools. We focus on enabling users to effectively access *all* the information contained in a complex document — not only the image portion, or only the text portion, or the only numerical portion.

To optimize the overall performance of the system we use an integrating principle of a probabilistic document model which integrates, in a principled manner, interpretations and reliability estimates thereof from all system components. In this way we intend to meliorate, if not overcome, the risk of error propagation. In addition, this approach enables "goal-directed interpretation" in which the process of responding to user queries can initiate reinterpretation of document image data.

The remainder of the paper describes our vision for useful analytic CDIP tools in more detail. In the following section, we describe issues in CDIP system design, including its functional requirements and system architecture. We then deal with the involved question of how to evaluate CDIP technologies in general, and our plans for the building the first extant complex document test corpus and how it will be used to evaluating and developing the system further. We conclude summarizing our current state of development and further issues for exploration.

II. CDIP SYSTEM DESIGN

A. Functional requirements

The focus is on developing a prototype tool suite implementing three core CDIP technologies of *text analysis, integrated retrieval, and data mining*. Specific attention is being paid to modular design, to ensure that

the developed software modules will easily integrate into different task-level applications.

Text analysis augments text documents and components with canonical class labels or structured interpretations. This eases querying, since the user can sometimes refer to labels from a finite set rather than anticipating all vocabulary variations. It also aids data mining, by providing sets of attributes that are smaller, less ambiguous, and more task specific than natural language words and phrases. We currently concentrate on two key text analysis technologies: text categorization [1] and stylistic analysis [2], [3].

Integrated retrieval from different kinds of data sources is a key mediator function—our IIT Mediator system currently supports a variety of traditional data sources such as unstructured text, semistructured XML/text data, as well as structured database querying [4]. A rule-based source selection algorithm selects those data sources most relevant to an information request, enabling the system to take full advantage of domain-specific searching techniques, such as translation of a natural language request into a structured SQL query. Results are then fused into an integrated retrieval set [5]. Such integrated domain-specific search is unique, to the best of our knowledge, in the current state of the art. The prototype builds upon this foundation to allow sophisticated querying of information extracted from complex documents at previous levels. Spatial relations between document elements, such as adjacency, are straightforward to encode and query, as are keywords denoting topic, style, and graphic categories. This will enable searching for clusters of related data elements, as in queries such as “Find three orders mentioning attacks on different countries, all signed by the same person.”

Data mining can leverage the fact that the IIT Mediator already indexes many different kinds of data using a relational database model, allowing straightforward integration of structured and unstructured data. We are developing tools that will augment this structure with stylistic, graphic classification, and template-based data. This approach allows application of traditional data mining and machine learning methods to discover relationships between different data such as association rules [6] and document clusters [7]. We will further develop routines to find correlations in document descriptors (for example, possible relationships between the author of a document and particular language styles).

These constitute relatively robust, broadly applicable technology sets for whose elements a number of existing implementations are available. They also produce outputs

particularly compatible with other components of the system. In the near future, we will also examine incorporation of richer, but more domain-specific, technologies such as named entity recognition and information extraction.

B. Processing Modules

The prototype architecture is designed as a generic framework for integrating component technologies with appropriate APIs and data format standards to allow ‘plugging in’ different subsystems for performing component tasks. We emphasize that, throughout, software development work in the prototype is on interfacing and integrating existing available components, rather than on developing new ones.

Image analysis in the prototype comprises modules for low-level image processing, which segment different components of a complex document. Among these processes, printed text is separated from handwriting tables, and graphics, signatures and logos are identified, characters and words in printed text are segmented from each other, and features characterizing these different components are extracted for higher-level processing. Additional modules are included to determine spatial relations between document image components for: geometric analysis of document structures, parsing of tables and lists, and finding geometric associations.

Symbol analysis consists of processes for extracting symbolic content and relations from processed image data. This symbolic content can serve as input to the information database and conceptual analysis functions which provide knowledge-level analyses (retrieval, data mining, reliability assessment) using these primitives. Symbol analysis covers a range of different functions, including letter/word identification (i.e., OCR), handwriting attribution, classification of maps and diagrams, and parsing of tables and lists in the document. While some of these functions are more language-based and others more graphically-based, in the system these processes will interact via a probabilistic document structure model.

High-level information access and analysis functions consist of information retrieval from a collection’s heterogeneous data and data mining to find interesting and meaningful patterns in that data. Retrieval in the prototype is based around IIT Mediator technology maintains a database of extracted information and handles user requests. The functionality at this level is integrated retrieval from heterogeneous data extracted from complex documents. As well, we are studying and

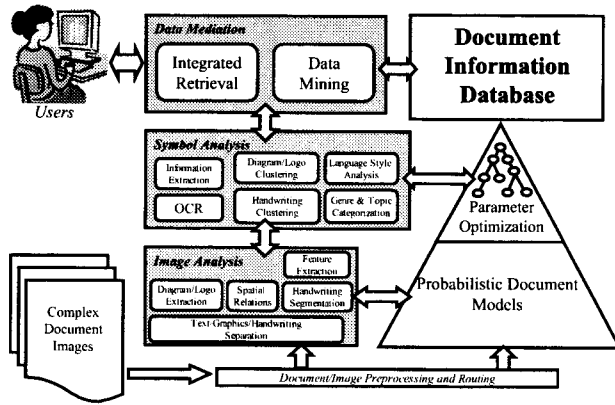


Fig. 1. Architecture of the full envisioned CDIP prototype.

working on techniques for pattern detection on complex document collections. Our main goals are: extracting information, finding relational and temporal links and patterns, and organizing this information via a usable interface. While such tasks have been addressed previously in other contexts, and some have made some use of speech recognition and OCR on news data, the challenges are much greater with diverse, complex material from multiple sources. For instance, it may be necessary to maintain multiple ambiguous interpretations in the output of handwriting recognition and OCR, which can stress data mining and pattern finding techniques. Thus as our project progresses, scalability in the face of large and ambiguous datasets will be a key ongoing consideration.

C. Software architecture

A key component of our approach is to tightly integrate the processes of document image interpretation, symbol extraction and relation, information retrieval, and data mining. An integrated approach promises increased reliability for all of these processes. Constraints on image interpretation, based on data consistency concerns at the high-level, can improve reliability of image interpretation. Similarly, gaps in the database can potentially be filled in at retrieval time, by reinterpreting image data using top-down expectations based on user queries. Our envisioned architecture for the complete system is given in Figure 1. It is divided into three main levels, each comprising several modules: *image processing*, which segments and extracts structures from images (for OCR we will integrate existing software in the obvious fashion), *symbol analysis*, which performs linguistic

and classification analysis of extracted information for annotation in the database, and *data mediation*, which includes integrated information retrieval and data mining using the complex document information database constructed by the first two levels.

Image processing for CDIP must separate out different image components from one another, prepare them for further processing (OCR and classification), and annotate geometric relations between them. Separation of text from graphics and handwriting, as well as geometric normalization of each component, can be done using directional and regulated morphological operations [8], [9]. Both global and local features of each component are extracted, for use by symbol analysis (see below). Spatial relations between components are classified and then used to construct a document structure model which can provide useful annotations in the document database (such as that certain components are likely titles, or logos, or signatures, for example). Key will be the use of knowledge-based constraints at this level to help constrain the interpretative processes. For example, probabilistic document structure models may be used which give probabilities of image-level features, conditioned on structural labels for image regions [10]. Dynamic programming can then be used to solve for a maximum likelihood structure under the model. If multiple possible document structures are allowed, the highest probability model will be chosen for each given document.

The main function of the symbol analysis layer is to annotate document component data (both graphic and textual) with meaningful metadata which enables more effective utilization at the mediation level. Diagram clustering [11] will provide symbolic labels for similar graphics in the corpus, allowing linking of different documents with the same symbols or logos. Manual annotation can easily be added as well to make such labels meaningful to human analysts. Similarly, handwriting and signature analysis will be used to identify the authors of handwritten notes (either by recognition based on known samples, or by clustering to determine that different samples are from the same hand). Too, the linguistic style of textual components (extracted by OCR) will be analyzed [2], [3] to determine a profile of the text and its author, for genre (command, report, request), tone (authoritative, submissive), and for author characteristics (gender, age, background). In all of these cases, symbolic labels will be inserted appropriately into the corpus database which can then be searched over, or used as data fields for data mining. In the near future, we also intend to incorporate more in-depth linguistic

processing including part-of-speech tagging [12], named-entity tagging (e.g., [13]), and information extraction (e.g., [14]), which will enable inclusion of more detailed structured data in the corpus database.

To support flexible querying of the complex document information at the level of *concept analysis*, the prototype will maintain a database of all the information extracted from the corpus (this also includes unstructured text). The IIT Mediator is built using the SIRE information retrieval infrastructure [15], [16] which represents all information, whether unstructured text, semistructured XML, or structured data, in a relational database format. This uniform substrate gives us a foundation to build upon to allow more sophisticated querying of information extracted from complex documents at previous levels. Spatial relations between document elements, such as adjacency, are straightforward to encode and query, as are defined keywords denoting topic, style, and graphic categories. In addition, this opens up larger possibilities for future work, of searching for clusters of related data elements, as in queries such as “Find three letters discussing corporate mergers, signed by the same person.”

Finally, our relational approach to data representation allows the application of traditional data mining and machine learning methods to discover new associations in the data, such as association rules [17] and document clustering [18], [19]. While our work on data mining from complex document collections is just beginning, we will work on developing algorithms that look for correlations in document descriptors (for example, possible relationships between the author of a document and particular language styles). Additionally, the relational structure will ensure that our approach can be combined with other external data (biographical data on authors, etc.) to give users a more complete picture.

In addition to all the above, the use of an architecture that integrates all CDIP modules, from image processing to information retrieval and data mining, will also enable top-down application of interpretative constraints based on user interaction and feedback. For example, if only partially relevant results are found for a user’s query, a further attempt may be made to find information of the requested type by re-examining the original documents in light of the particular type of information requested (and what is currently known).

III. DATA SETS AND EVALUATION

A central task for a CDIP project is evaluating the effectiveness of the system, and how that effectiveness

responds to changes in effectiveness of components (OCR, IR, table and diagram extraction, etc.). A major tool in evaluation are test collections, i.e. data sets manually annotated with desired or correct interpretations.

Test collections have played a large role in the evolution of the fields on which CDIP depends: document analysis, information retrieval, computational linguistics, and data mining. Test collections are used both to evaluate technologies in these areas, and to improve the effectiveness of these technologies through tuning parameters and fitting statistical models. Indeed, much recent progress in these fields has resulted from the incorporation of new statistical and machine learning methods enabled by the availability of labeled data sets.

In contrast, we believe progress on unified CDIP systems has lagged precisely because of the lack of a high quality, publicly available test collection. The construction of such a collection is a major goal of our project. We describe our progress in this area below, along with how we will use the resulting collection in evaluating capabilities of our CDIP system.

A. Document Collections for CDIP Evaluation

A good test collection for CDIP should cover the richness of inputs a CDIP system may face: a range of document formats, structures, sizes, and genres, manifested with varying print and imaging qualities. Documents should include handwritten text and annotations, diverse fonts, and elements such as images, tables, logos, and other graphic complexities. The volume of documents, and the number of redundant or useless documents, should be large enough to stress the component technologies and the system as a whole. Further, given our interest in intelligence applications, we want our collection to contain private communications within and between groups of people planning activities and deploying resources. Finally, the data in the collection should be publicly available to researchers with minimal costs and licensing restrictions.

Documents that become public through legal duress (lawsuits, criminal trials, FOIA requests, etc.) provide one of the best sources of such material. We have chosen to base our collection on the Legacy Tobacco Documents Library, a collection of approximately 7 million documents (42 million pages) that became public through legal proceedings against US tobacco companies. We have arranged to obtain these documents in TIFF format, with associated metadata, from the University of California - San Francisco [20].

Besides satisfying the criteria we listed in the previous section, the tobacco documents have the advantage of an associated active research community. More than 200 peer-reviewed papers have been published using documents from LTDL and similar sources [21], and the US National Cancer Institute has funded more than a dozen groups to study various topics using tobacco documents [22]. We plan to leverage this interest to aid CDIP evaluation, as discussed in the next section.

B. Methods and measures

Evaluation of the CDIP prototype and its components requires not just documents, but also defined tasks, desired system outputs for those tasks, and measures of how well the system produces such outputs. We are defining such evaluation tasks for three key aspects of our prototype's performance: document analysis, information retrieval, and data mining.

1) *Document Analysis*: One set of evaluations will measure the effectiveness of our document analysis components. These results will help direct future research and development by both: (a) aiding evaluation of which component algorithms perform best on complex documents, and (b) enabling us to determine which (if any) document analysis components are responsible for retrieval or mining errors at the high-level. Testbed tasks will be constructed by manually annotating samples of 50-100 diverse document images for each type of desired output: text segments, document structure, diagram tags and relations, recognized characters, and so on. We are making initial use of a stratified sample of about 1200 tobacco documents produced by tobacco document researchers at University of Georgia [23]. This sample has been annotated with ground truth for printed and handwritten text.

Outputs generated by CDIP components will be compared to the manual ground truth and difference recorded. Standard effectiveness measures (e.g. character- and word-level error rates for OCR) will be used.

2) *Information Retrieval*: Test collections for information retrieval require, in addition to documents, a set of *topics* (specifications of certain documents of interest, and queries a user might enter to find them) and corresponding *relevance judgments* (lists of which documents should be retrieved for each topic). We are in discussions with several tobacco document groups to formalize as topics some of the subject areas in which they are searching for documents, and to cooperate on producing relevance judgments.

While these efforts are at an early stage, several aspects of the topics already suggest they will provide a good challenge for CDIP capabilities. Many topics specify not only documents on a particular topic, but documents of a particular genre (e.g. memos or minutes) or with a particular audience (internal vs. external communications). Such topics provide a challenge to both document image analysis and linguistic stylistic analysis. Handwritten documents and annotations are also of particular interest, raising the question of what can be done to improve their accessibility short of (currently impossible) arbitrary handwriting recognition.

We will measure the ability of the CDIP prototype to retrieve relevant documents in response to queries for each topic, and compute standard effectiveness measures such as simple average precision.

3) *Data Mining*: Another type of processing that we will support is a full scan of a set of complex documents to look for *interesting* patterns. Evaluation of data mining will focus on both effectiveness and efficiency. Effectiveness will be measured against small amounts of manually annotated data. We will investigate active learning methods in producing annotated data for training and experimental design methods for reducing the amount of data to annotated in order to determine differences among systems. In some cases, it may be possible to use one part of a complex data set as ground truth for pattern finding on another. For example, one document may list the membership of a group at a particular time, while determining that from other documents would be of interest. Tobacco document researchers have also established ground truth for some items of interest, e.g. links between various tobacco industry personnel and outside organizations.

Efficiency is an issue not just because of the size of complex document collections, but because of the ambiguity introduced by OCR, handwriting recognition, and document structure analysis. This ambiguity can vastly increase the range of patterns which a data mining system must consider, impacting running time and memory (as well as accuracy). We will focus in particular on how algorithms for association rules and temporal sequence patterns scale with the degree of ambiguity in interpretation.

IV. CURRENT STATUS

A. The IIT Mediator

The current prototype of the IIT Mediator¹ implements the essential functionality of integrated retrieval we have described, for document data represented in a structured format (as above). It deals in a seamless manner with a variety of data sources, including free-text, semistructured XML, and structured databases (complex document information will be integrated soon, as lower-level processing and extraction modules reach maturity). The IIT Mediator keeps separate stores of metadata for each type of source that it has access to, which enables it to determine which sources are most likely to have data relevant to a given query. Simple natural-language query processing extracts functional roles from the query that a rule-based processor then translates into source-specific queries. Results are then integrated via an information-fusion algorithm [5]. Currently, the Mediator sends the query in its natural-language form to all available unstructured sources, essentially performing a metasearch. The idea is that the Mediator results, in addition to possibly containing an answer to the user's query, will at least be no worse than those obtained from a traditional metasearch [24].

B. Symbolic document processing

For symbolic analysis, we have focused to date primarily on topic and style based text categorization. Work on these tasks requires solving two main problems: developing algorithms to build accurate classification models, and constructing sets of features that are both useful for text categorization and computable. For the former we have both used existing machine learning methods, as well as a novel text categorization method based on Bayesian logistic regression [25]. We have also developed a comprehensive infrastructure for processing and representing linguistic annotations for texts in a database format, which includes tools for extracting a wide variety of linguistic textual features. We have further developed resources for features based on Systemic Functional Linguistics [26], which enable more accurate and more insightful stylistic text classification, in many cases [27], [28].

C. Image processing

The image processing components of our system will rely on commercial software components as well as additional components we are developing. Commercial

OCR engines include in addition to OCR/ICR additional processing capabilities that are necessary for document analysis. These capabilities include noise filtering and skew correction, adaptive image segmentation, detection of page orientation, and document zone determination. Detected zones include text blocks, tables, and graphics/pictures. The performance of commercial engines on noisy document images is yet to be determined. In addition to a commercial OCR engine we will be using the *DocLib* system developed by the Language and Media Processing Laboratory of the University of Maryland, as an integration framework. DocLib includes in addition to low-level processing capabilities, several high level-modules such as logo-detection and script-identification. The components we are developing for this system include document image de-warping modules [29], [30], separation of text from handwriting using directional regulated morphological operations [8], [9], signature clustering, and table identification in noisy documents.

V. SUMMARY

Complex document information processing is of increasing importance to intelligence analysis in the modern world, and yet little work has been done on developing integrated solutions to this problem. We have laid out an integrated architecture which unifies the various component technologies, giving a clear roadmap for research on developing useful CDIP systems.

Acknowledgments

This work is currently supported by a Challenge Workshop grant from ARDA.

REFERENCES

- [1] D. Lewis, R. E. Schapire, J. P. Callan, and R. Papka, "Training algorithms for linear text classifiers," in *SIGIR '96: Proc. of the 19th Int. Conference on Research and Development in Information Retrieval*, 1996.
- [2] S. Argamon, M. Šarić, and S. S. Stein, "Style mining of electronic messages for multiple author discrimination," in *Proc. ACM Conference on Knowledge Discovery and Data Mining*, 2003.
- [3] S. Argamon, M. Koppel, J. Fine, and A. R. Shimony, "Gender, genre, and writing style in formal written texts," *Text*, 2003.
- [4] D. Grossman, S. Beitzel, E. Jensen, , and O.Frieder, "IIT Intranet Mediator: Bringing data together on a corporate intranet," *IEEE IT Professional*, vol. 4, no. 1, pp. 49–54, 2002.
- [5] S. Beitzel, E. Jensen, A. Chowdhury, D. Grossman, O. Frieder, and N. Goharian, "On fusion of effective retrieval strategies in the same information retrieval system," *Journal of the American Society of Information Science and Technology*, vol. 55, no. 10, 2004.

¹Demo version available at <http://www.ir.iit.edu/mediator>.

- [6] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo, "Fast discovery of association rules," in *Advances in knowledge discovery and data mining*. American Association for Artificial Intelligence, 1996, pp. 307–328.
- [7] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *Proc. SIGKDD Workshop on Text Mining*, 2000.
- [8] G. Agam and I. Dinstein, "Adaptive directional morphology with application to document analysis," in *Mathematical Morphology and its Applications to Image and Signal Processing*, P. Maragos, R. W. Schafer, and M. A. Butt, Eds. Kluwer Academic Publishers, 1996, pp. 401–408.
- [9] —, "Regulated morphological operations," *Pattern Recognition*, vol. 32, no. 6, pp. 947–971, May 1999.
- [10] J. Liang, I. Phillips, and R. Haralick, "A methodology for document image structures extraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 719–734, 2001.
- [11] S. Aksoy and R. M. Haralick, "Graph-theoretic clustering for image grouping and retrieval," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1999, pp. 63–69.
- [12] E. Brill, "Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging," *Computational Linguistics*, vol. 21, no. 4, pp. 543–565, 1995.
- [13] X. Carreras, L. Marques, and L. Padro, "Named entity extraction using adaboost," in *Proceedings of CoNLL-2002*, Taipei, Taiwan, 2002, p. 167170.
- [14] D. Freitag, "Machine learning for information extraction in informal domains," *Machine Learning*, vol. 39, 2000.
- [15] O. Frieder, A. Chowdhury, D. Grossman, and M. C. McCabe, "On the integration of structured data and text: A review of the SIRE architecture," in *DELOS Workshop on Information Seeking, Searching, and Querying in Digital Libraries*, Zurich, Switzerland, December 2000.
- [16] C. Lundquist, O. Frieder, D. Holmes, and D. Grossman, "Integrating structured data and text: A relational approach," *Journal of the American Society for Information Science*, vol. 50, no. 5, April 1999.
- [17] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. Morgan Kaufmann, 12–15 1994, pp. 487–499.
- [18] A. Ruocco and O. Frieder, "Clustering and classification of large document bases in a parallel environment," *Journal of the American Society of Information Science*, vol. 48, no. 10, October 1997.
- [19] —, "Parallel document clustering process. US Patent #5,864,855. Patent filing by the Office of the Judge Advocate General, Department of the Army." January 1999.
- [20] H. Schmidt, K. Butter, and C. Rider, "Building digital tobacco industry document libraries at the university of california, san francisco library/center for knowledge management," *D-Lib Magazine*, vol. 8, no. 2, 2002. [Online]. Available: <http://www.dlib.org/dlib/september02/schmidt/09schmidt.html>
- [21] N. Hirschhorn, "Research reports and publications based on tobacco industry documents, 1991-2005," May 2005. [Online]. Available: http://tobacco.health.usyd.edu.au/site/gateway/docs/Hirschhorn_publications_list.pdf
- [22] "Review and analysis of tobacco industry documents," June 1999, national Cancer Institute Program Announcement. [Online]. Available: <http://grants.nih.gov/grants/guide/pa-files/PAR-99-114.html>
- [23] W. Kretzschmar, C. Darwin, C. Brown, D. Rubin, and D. Biber, "Looking for the smoking gun: Principled sampling in creating the tobacco industry documents corpus," *Journal of English Linguistics*, vol. 32, pp. 31–47, 2004.
- [24] D. Dreilinger and A. E. Howe, "Experiences with selecting search engines using metasearch," *ACM Transactions on Information Systems*, vol. 15, no. 3, pp. 195–222, 1997.
- [25] A. Genkin, D. D. Lewis, and D. Madigan, "Large-scale bayesian logistic regression for text categorization," *submitted*, 2004.
- [26] M. A. K. Halliday, *Introduction to Functional Grammar*, 2nd ed. Edward Arnold, 1994.
- [27] S. Argamon and J. Dodick, "Conjunction and modal assessment in genre classification," in *Notes of AAAI Spring Symposium on Attitude and Affect in Text*, March 2004.
- [28] C. Whitelaw, N. Garg, and S. Argamon, "Using appraisal taxonomies for sentiment analysis," *submitted*, 2005.
- [29] G. Agam and C. Wu, "Structural rectification of non-planar document images: application to graphics recognition," in *Graphics Recognition: Algorithms and Applications*, ser. LNCS, D. Blostein and Y.-B. Kwon, Eds., vol. 2390, 2002, pp. 289–298.
- [30] C. Wu and G. Agam, "Document image de-warping for text/graphics recognition," in *Structural, Syntactic, and Statistical Pattern Recognition*, ser. LNCS, T. Caelli, A. Amin, R. Duin, M. Kamel, and D. de Ridder, Eds., vol. 2396, 2002, pp. 348–357.

DOCLIB : A Document Processing Research Tool

Kevin Chen

Booz | Allen | Hamilton
134 National Business Pkwy
Annapolis Junction, MD 20701
chen_kevin@bah.com

**Stefan Jaeger
Guangyu Zhu
David Doermann**

Institute for Advanced Computer Studies
University of Maryland,
College Park, MD 20742, USA
jaeger@umiacs.umd.edu

Abstract

Often, valuable document processing intellectual capital is lost due to staff transitions or project restructuring prior to technology transfer. Furthermore, hardware and software integrity, dependencies, and compatibility are critical components that often impede technology migration. While many open source tools attempt to mitigate these issues, they do not always address specific design needs and tailored-process that Government organizations must adhere to. This paper addresses the need for a common document processing research vehicle through which institutions can develop and share research-related software and applications across academic, business, and Government domains.

1 Introduction

Although research does not generally commission the development of production quality software, the absence of a software development framework and guiding process can significantly impede progress upon the migration of technologies. The transfer of technology across organizational boundaries requires a significant time investment. These efforts are often exacerbated when there is incomplete documentation, unpredictable methods required to build and execute software, inconsistent software dependencies, and/or a misuse of software. The setup, configuration, and execution of research-related software can be difficult without a structured and shared development and delivery process. Although a time investment is required during software acquisition, a lack of infrastructure and process causes an avoidable loss of intellectual capital within the document processing community. A set of core research tools developed with an eye toward production, quality, and stability can allow valuable intellectual resources to be more focused on creative and innovative document processing solutions. This, in turn, can maximize return on investment (ROI) for funding organizations. This paper prescribes a systematic development approach and shared architecture which has been successfully used in collaboration across academia, business, and

Government environments. The DOCLIB Collaboration Model also includes an “add-on” construct that allows research related applications to be developed and shared.

The DOCLIB Collaboration Model has been effectively used as a communication portal between University of Maryland, the Department of Defense, and industry. The use of this collaboration model has resulted in a shared document image processing C++ toolkit termed core-DOCLIB. This toolkit is currently funded by a specific Government division.

Most existing software packages satisfy only a subset of the document processing user base and do not prescribe a robust development process to facilitate technology transfer [11,12,13,14]. For instance, Intel’s OpenCV Toolkit provides high-level computer vision methods for Windows users only [11]. This limitation creates a barrier across the enterprise and discourages collaboration. Alternatively, MATLAB does not provide the ease-of-use and efficiencies needed for a collaborative model [14]. The core-DOCLIB toolkit, however, supports various platforms (e.g.: Windows, Solaris, Linux, etc) and compilers (e.g.: c++.net, gcc, g++, Forte) and readily conforms to security prerequisites most Government agencies must adhere to. Moreover, the external API was designed to be easy-to-use for those users new to the object oriented world.

Our intention is to advertise the DOCLIB collaboration model across varying Government divisions. Decisions relating to sharing of the core-DOCLIB software, shared development environment, and add-ons are in progress.

2 Approach

Compliance to a structured development process that directly addresses many of the technology migration obstacles currently facing the industry was the driving force behind the core-DOCLIB success. The DOCLIB Collaboration Model involves the use of the core-DOCLIB toolkit and the adaptation of a shared process used to foster collaboration.

Core-DOCLIB is a C++ toolkit that provides document/image-processing capabilities through a documented, easy to use interface. This library seeks to provide a functional, stable, and robust environment that supports the collaborative development of research capabilities. Use of the DOCLIB Collaboration Model has facilitated technology migration by mitigating debugging time, number of code reviews, and configuration management challenges.

Document processing research applications that are built on top of the core-DOCLIB library are referred to as DOCLIB add-ons. A DOCLIB add-on standardizes the delivery and use of research applications. Robust test scripts, standard naming conventions, auto-generated documentation, and an organized directory hierarchy are some of attributes that are included as part of an add-on. Adherence to the add-on conventions fosters discipline within organizations resulting in less time-consuming technology migration efforts. The application of the DOCLIB Collaboration Model to 'add-ons' has improved collaboration efforts between the Department of Defense and the University of Maryland over the past two years.

3 Core-DOCLIB

The establishment and use of the DOCLIB Collaboration Model fueled the creation of the core-DOCLIB C++ toolkit. This toolkit has served to satisfy document processing requirements set forth by academia and the Department of Defense. It was necessary for core-DOCLIB to be scalable, flexible, and extendable to meet specific organizational needs. For example, use of an Image Factory was introduced in the core-DOCLIB architecture to satisfy a need for plug-and-play image formats. This design allows image types to be added or removed without modifications to the existing core-DOCLIB code.

DLImage DLImage is the centralized image object used to store image data and header information when loading an image. The DLImage object is a generic image format that is independent of the input image types (i.e. TIFF, JPEG, GIF, etc.). DOCLIB Image processing algorithms are designed as independent modules that can be added or deleted without affecting other core-DOCLIB features (See Figure 1).

Core-DOCLIB currently supports the following image processing features:

- Resize an image
- Rotate an image
- Subimage
- Flip an image
- Contour an image
- Reverse black/white pixels on an image
- Paste an image
- Project
- Calculate connected components
- Draw Line, Box, Pixels onto an image
- Dilate an image
- Erode an image
- Sharpen an image
- Blur an image
- Conversions:
 - 256 Color quantized
 - Percent Thresh hold Binarization
 - Threshold Binarization
 - Color To Gray
 - Threshold Gray To Binary
 - NIBlackBinarization
 - Binary To Color
 - Gray To Color
 - Binary To Gray
 - YCrCb Binarization
- Skeletonize
- Deskew
- Centroid

Image Factory This factory inherits the Factory Design Pattern concept and extends the factory concept (See Figure 2). The Image Factory enables a higher lever of abstraction, encapsulating the implementation of all image objects. Moreover, the Image Factory provides the logic to determine an image's file format without the user specifying.

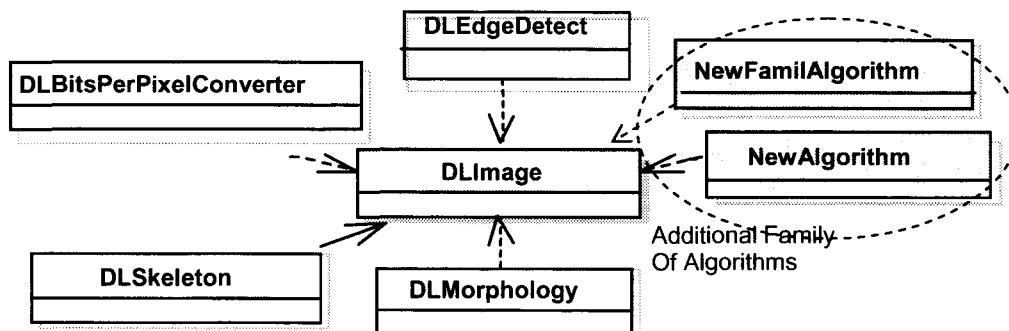


Figure 1: DOCLIB Algorithm Hierarchy

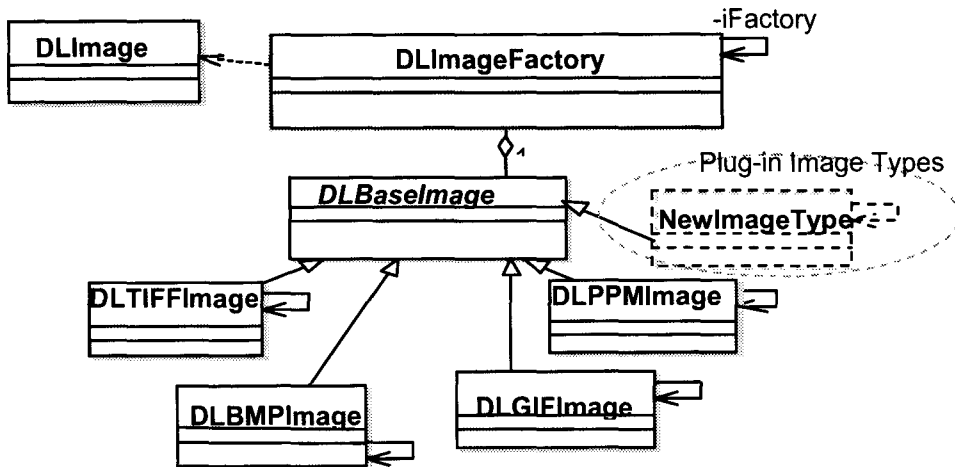


Figure 2: Image Factory class diagram

All image type objects are declared statically, therefore, the image type objects are constructed when the application starts. Image type objects register their existence at the Image Factory during construction (see Figure 3). Once the new image object has been registered with the Image Factory, the new image type will be one of the supported images in core-DOCLIB. The core-DOCLIB Image Factory architecture has been proven and allows the Department of Defense to ‘plug-in’ proprietary image types into core- DOCLIB without any code modification.

cognitive level than just the image pixels, such as ground-truth data for classifier training or logical segmentations into lines or characters. DLImage and DLDocument are two independent modules standing side by side and interacting with each other, as illustrated in Figure 4. In particular, DLDocument and its subclasses have access to their own image data and can perform basic image operations using the methods in the DLImage class.

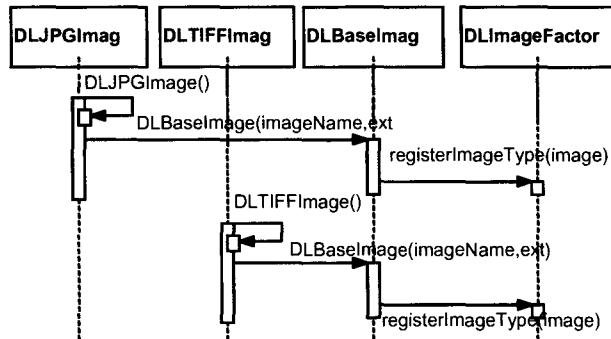


Figure 3: Image plug-in registration process

3.1 DLDocument

The DLDocument class was defined as a container for the results of basic image operations and metadata calculations performed by other DOCLIB classes and add-ons. While DLImage is a container for the basic image operations implemented in DOCLIB, DLDocument offers data structures and methods for manipulation of metadata, i.e. data that is on a higher

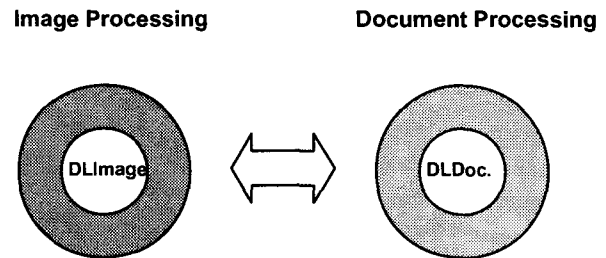


Figure 4: DLImage and DLDocument

DOCLIB’s DLDocument class understands documents as hierarchical entities composed of pages containing regions of interest, which we call zones. Accordingly, DOCLIB’s document hierarchy consists of the following objects:

- DLDocument
- DLPage
- DLZone

The zones are hierarchical structures in themselves and can contain sub zones (see Figure 5).

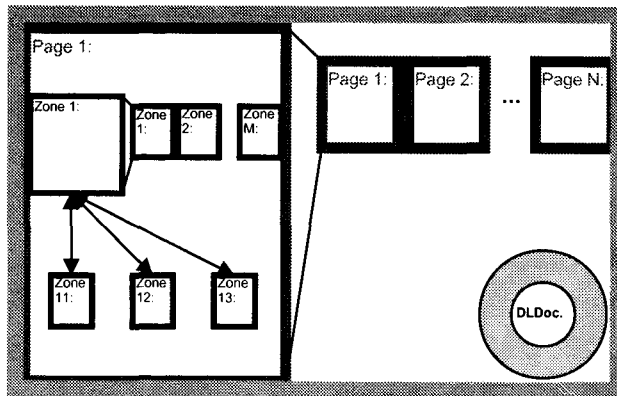


Figure 5: DOCLIB's document hierarchy

The DLDocument class represents the highest level in the DOCLIB document hierarchy. It is a container for attributes that describe the layout and content of one or more pages within a document. Each DLDocument object has a unique document ID and features a list of document tags allowing developers to dynamically append or remove additional self-defined attributes. Several constructors are available, depending on the information present at the time of construction. The pages of a document are stored in a simple list structure. Developers have access to this list via a set of functions for appending, inserting, or deleting either one single page or a list of pages.

The DLPage class represents the metadata and logical content of a document page. Its main purpose is to accommodate information specific to pages rather than whole documents. The structure of DLPage, however, is similar to DLDocument. It contains a page ID and a dynamic attribute list that developers can use to store self-defined attributes. The main difference to the DLDocument class is a list of zones, which are general containers defining specific regions on a page, such as text blocks, images etc. Similar to DLDocument, DLPage provides several functions for appending, inserting, or deleting zones on a page. In addition, it provides methods for accessing page-related information, e.g. its spatial dimensions or the corresponding DLImage.

A DLZone object is a general container for objects on a document page. DLZone offers a general bounding box concept that allows zones to have sub zones, so-called child zones. These child zones are stored in the same list structure used by DLDocument and DLPage to store pages and zones, respectively. A zone and its sub zones span a dynamic tree structure in which zones (nodes) can easily be appended, inserted, or deleted. In addition, DLZone provides methods for merging and splitting zones. Similar to methods in DLPage, DLZone offers methods for accessing information specific to zones. A Zone object is basically defined by its origin, width and height. DOCLIB allows developers to dynamically change this information. Each DLZone

object automatically performs a consistency check whenever the developer tries to do so, and denies the operation when it would lead to an inconsistent state, for instance when a zone would extend the boundaries of its parent.

In order to allow programmers to adapt DOCLIB to their own needs, and customize DOCLIB objects to the requirements of a specific application domain, developers can subclass DLZone and augment it with more specific information. For example, typical user-defined subclasses of DLZone could be DLTextLine and/or DLCharacter etc.

4 Software Development Process

The DOCLIB Collaboration Model has shown that adapting a structured development process assists throughout the technology migration process. Several well-defined practices were established and integrated as part of the development and release proceedings.

Distribution portal Coordinating software changes or enhancements without a distribution mechanism can be quite challenging. Given the complexity of document processing related research, it is often that bug fixes/code enhancements or modifications are transferred to end-users. A web portal that allows users to check status, download specific versions of the libraries, and seek help is invaluable to coordinating and communicating during technology transfer.

Source control One of the most important aspects of collaborative code development is the sharing of the code itself. Without all users having access to the same code base, each party will inherently develop similar but different baselines in parallel. Several open source version control software packages facilitate the management of software development initiatives. These systems provide developers with the ability to obtain and modify source code for their projects. If errors occur, a configuration Management (CM) tool provides methods for reverting code to a previous known stable state.

Documentation standards Thorough and understandable documentation is crucial to the success of many technology transfer efforts. Unfortunately, it is also the most often ignored. Regimented documentation standards and use of auto-generating documentation packages (ie. Doxygen [7]) allow user documentation to be quickly and easily created. This information is often the single source customers will use to gain insight into the operation and functionality of software.

Bug tracking Bug tracking helps to ensure the development of quality software products. Bug tracking gives users and developers the ability to document

bugs, request new features, and check the status of the resolutions to their issues. We use the popular Mantis bugtracker for DOCLIB [10].

Software integrity It is difficult to guarantee software integrity within the document processing research field given the complexity of the science and the variability in data. The need to verify system behavior was evident early on during the development of core-DOCLIB. A regression test application was prescribed that thoroughly tested all behavior within the system. This module is often used by software developers before committing changes to existing code. This approach has also proven invaluable in identifying obscure bugs before formal core-DOCLIB software releases.

The execution of regression tests also facilitates the observance of memory leaks throughout the system. Advertisement of the regression tests has helped build buy-in and commitment to customers of core-DOCLIB and/or add-on modules. This mechanism has proved very useful in minimizing the number of bugs occurring throughout core-DOCLIB.

5 DOCLIB Add-Ons

DOCLIB has proven its success as an underlying image processing library in the DoD community and at University of Maryland for the last two years. DOCLIB not only helps resolve compatibility issues among different research groups but also helps to reduce the amount of code that needs to be imported into the Government workspace.

The core of DOCLIB provides basic image processing methods that have been successfully tested in numerous practical applications and are considered standard by researchers and programmers alike. Add-on modules, however, are solutions to problems that have hitherto eluded practical approaches, and are still actively investigated within the research community. Methods competing with other techniques are also implemented as add-ons, either because the research community has not yet given the final verdict on their underlying theory or they are fine-tuned to specific problem types and can thus not be considered standard for the general case. DOCLIB provides most of its document processing functionality as add-ons to the DOCLIB core. Table 1 of the appendix lists some of the currently available add-ons for DOCLIB.

6 Conclusion

A shared software development toolkit and structured development process can significantly enhance technology migration experiences within the document image processing community. The DOCLIB Collaboration Model prescribes a methodology and underlying toolkit to satisfy these needs.

Recipients of DOCLIB add-ons can be reassured that a structured development process was adhered to.

Moreover, the location of documentation, source code, test scripts, etc will be consistent regardless of the organization developing it. Once an organization is a user of core-DOCLIB, only add-on related code will have to be transferred to receiving organizations.

The DOCLIB Collaboration Model has been effectively used as a communication portal between University of Maryland, the Department of Defense, and industry. The use of this collaboration model has resulted in a shared document image processing C++ toolkit termed core-DOCLIB. This toolkit is currently funded by a specific Government division. Our intention is to advertise this collaboration model across varying Government divisions. Decisions relating to sharing of the DOCLIB software, shared development environment, and add-ons are in process.

Acknowledgments

The partial support of this research under DOD contract MDA90402C0406 is gratefully acknowledged.

References

- [1] 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*, Vol. 19, No. 2, pp. 176–181, Feb. 1997.
- [2] L. O’Gorman, The Document Spectrum for Page Layout Analysis, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 11, pp. 1162-1173, Nov. 1993.
- [3] S. Jaeger, Informational Classifier Fusion, 17th Int. Conf. on Pattern Recognition (Cambridge, UK), pp. 216-219, 2004.
- [4] L. Golebiowski, Automated Layout Recognition, Symposium on Document Image Understanding Technology, Greenbelt (Maryland), pp. 219-228, 2003.
- [5] 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, 2000.
- [6] R. Ziemer, Removing Repetitious Horizontal Background Lines, Internal Report, BAH, 2004.
- [7] <http://www.doxygen.org>
- [8] G. Zhu, S. Jaeger, D. Doermann, A robust stamp detection framework on degraded documents, Proc. of SPIE Conf. on Document Recognition and Retrieval XIII, 2006, to appear.
- [9] G. Zi, Groundtruth Generation and Document Image Degradation, Technical Report: LAMP-TR-121/CAR-TR-1008/CS-TR-4699/UMIACS-

- TR-2005-08, University of Maryland, College Park, May 2005
- [10] <http://www.mantisbt.org>
- [11] www.intel.com/technology/computing/opencv/
- [12] C. H. Lee, T. Kanungo. The Architecture of TRUEVIZ: A groundTRUth/metadata Editing and VISualiZing toolkit. Technical Report: LAMP-TR-062/CS-TR-4212/CAR-TR-959, University of Maryland, College Park, February 2001.
- [13] T. Kanungo, C. Lee, J. Czorapinski and I. Bella, TRUEVIZ: A groundTRUth/metadata Editing and VISualiZing Toolkit for OCR, *SPIE Conference on Document Recognition and Retrieval*, 2001.
- [14] <http://www.mathworks.com/>

Appendix

Table 1: DOCLIB Add-ons

Application Type	Add-on Module Name	Description
Script Identification	ScriptIdentification	The DOCLIB script identification add-on performs script identification for machine printed document images. It is a modified version of the Hochberg algorithm described in [1]. It allows classifying a document image as being printed in one of the following scripts: Amharic, Arabic, Armenian, Burmese, Chinese, Cyrillic, Devanagari, Greek, Hebrew, Japanese, Korean, Latin, or Thai. Script Identification can also be retrained to focus on different language mixes.
Layout Analysis	PageLayoutDOCLIB	The page layout add-on provides an automated means of training a classifier to recognize a document layout or set of layouts. The classifier can then be used to score an unknown image. The software is easily modified to include other types of object information. More details are given in [4].
Page Segmentation	DLDocstrumDOCLIB	For page segmentation, DOCLIB provides a module implementing the Docstrum method for structural page layout analysis [2]. The Docstrum method is based on bottom-up, nearest-neighbor clustering of page components. It detects text lines and text blocks, and is advantageous over many other methods in three main ways: independence from skew angle, independence from different text spacings, and the ability to process local regions of different text orientations within the same image.
Line Processing	DLGetLinesDOCLIB	Given a deskewed, oriented text image with dark machine print text on a light background, the line detection add-on will calculate text line information based on the connected component information. The image must be deskewed and oriented in order to receive meaningful results. The algorithm utilized has proven robust given text images with low resolution, broken characters, connected scripts (e.g. Arabic), multi-component character scripts (e.g. Chinese), and noise.

Application Type	Name	Description
	ZRemoveLinesDOCLIB	The line removal module is used to read in a document image and locate the background horizontal lines [6]. It then creates an output file in which the background lines are removed while the text remains intact. This add-on is designed to work on handwritten documents where the user writes on lined paper. Consistent background lines with inconsistent writing are expected. Less than optimal (though still useful) output may result when this is not the case. Various parameters are defined in the module which, when modified, may improve performance depending on the data. The module does not rely on color information. Hence it is particularly handy when the background horizontal lines are similar to the color of the handwriting. The algorithm requires binary input images that are reasonably deskewed (roughly, between -9 and 9 degrees, inclusive).
Object Recognition	logoTokenMatchDOCLIB	This module implements a logo recognition algorithm. Given a potential logo it performs a gray scale correlation against a set of candidate logos. For each candidate a score between 0 and 100 is generated corresponding to the degree of similarity. The best match is provided along with the score. In order to improve performance, the algorithm stops comparing against candidate logos when the best score is beneath a pre-defined threshold.
	StampDetection	The stamp detection module detects and locates elliptic stamps or seals on an input image by identifying and characterizing elliptic connected edges using both their magnitude and orientation information [8]. For a given document, it outputs an image list of detected stamps together with their corresponding confidence values. The module shows good performance for images with moderate noise and can be efficiently used in batch mode to process a large set of images.
Classifier Combination	InformationFusion	This add-on is an implementation of a new technique for information fusion and classifier combination, developed at the University of Maryland for hard and noisy environments. The underlying information-theoretical idea is to first normalize the confidence values provided by different methods for the same problem, before actually combining them. The normalization is performed in such a way that each confidence value matches its informational content. The recognition candidate providing the maximum information over all methods will then be selected as the most likely candidate for a given classification problem and presented test pattern. The method has already been successfully applied to several document processing tasks, including character recognition and script identification. Several papers describing the theoretical background have been published [3].

Application Type	Name	Description
Degradation	DLDegradationDOCLIB	DOCLIB's degradation tool degrades images with different types of noise [9]. Its main purpose is to emulate noise typically introduced during the scanning of paper documents. The generation of well-defined noisy environments allows adaptation of recognizers to real-world noise usually encountered in practical documents. For instance, the following types of noise may be added to a document image: blur, jitter, horizontal and vertical lines, rotation, speckle, pixel flips, etc.
Performance Evaluation	F_ScoreDOCLIB	The F-Score module is used to compare two object sets, the putative set and the truth set. Based on this comparison, the precision, recall, and F-measure values are calculated along with the F-score(s) [5].

A Process Flow for Realizing High Accuracy for OCR Text

Kazem Taghva and Julie Borsack and Tom Nartker

Information Science Research Institute

University of Nevada, Las Vegas

Las Vegas, NV 89154-4021

Abstract

1 Introduction

The Information Science Research Institute (ISRI) has performed in-depth research of optical character recognition (OCR), information retrieval, and related topics since 1989. ISRI ran many experiments comparing OCR output to manually corrected versions of the same collections. Although our experiments have shown that average precision and recall are not negatively affected by OCR errors, we also identified and characterized problems that OCR text may cause when applied in retrieval and other information access tasks.

In nearly all large hard copy document conversion tasks, information retrieval is just one aspect of the entire project. For example, classification of documents and extraction of specific information may also be required of the collection. Therefore, the required level of accuracy for these information access tasks is an important consideration. Measuring OCR text accuracy though is a formidable undertaking without automation.

In this paper we present a means to automatically measure OCR quality of documents. We also recommend a hardcopy document conversion process that will perform automated OCR correction, determine a document's OCR quality, and provide a correction tool that can guarantee a certain level of accuracy for converted collections.

2 OCR and Information Access

Although most information today is created electronically, compiling a collection of documents generated from various text processing applications proves to be more difficult than one would expect. In fact, most large collections are produced using imaging and optical character recognition. These processes retain "the original" document form (the images) and create a consistent and automated method for data capture. Unfortunately, OCR generated

text can cause difficulties for information access (IA). Two widely applied IA tasks are information retrieval (IR)[4, 6, 10] and information extraction (IE)[3]. In this section we introduce a few of the difficulties we have encountered in our research when using OCR text with these technologies. We also note the level of accuracy we believe is required to perform these tasks effectively.

In Sections 3 and 4 we present two technologies that automate OCR correction and document quality control (QC). In Section 5, we introduce an additional semi-automated technology that can be used to further improve document quality if necessary. The application of these components after OCR will greatly improve the collection for any subsequent IA tasks.

2.1 Problems for Information Retrieval

ISRI studies on effects of OCR errors on retrieval have pointed out that certain advanced functionalities of information retrieval systems, such as ranking, are not robust enough to overcome OCR errors. It is our view that more advanced retrieval techniques and applications require a higher accuracy rate. For example, increased overhead in the index (especially for large collections)[16, 14, 20], potential unpredictability in document ranking[14, 15] and the unretrievability of poor quality documents[15, 17] are issues that should be considered.

We have shown that using automated post-processing can improve document quality with regard to these issues and in turn improve retrieval effectiveness[8]. Although the required level of accuracy may vary from application to application, we found that a 95% *word accuracy* should be the minimum for document collections with an average page length of ten pages or greater to attain results similar to what one would expect from a clean collection.[11]. If short abstracts are to be indexed, there may not be enough redundancy to compensate for

misrecognized words. A higher word accuracy rate may be needed to retrieve short documents.

2.2 Problems for Extraction

Information extraction is even more heavily affected by poor OCR. Unlike IR, where redundancy compensates for poor recognition, any misrecognized characters in the data to be extracted will cause an instance of the targeted data to be missed. Another issue for IE that usually plays no role for IR is zoning errors. The more complex a document structure becomes, the more difficult it is for an OCR to determine reading order. When reading order is incorrect or if the zone type is misconstrued, zoning errors are the result. Zoning errors affect IE by eliminating the logical order expected by an extraction algorithm. So not only does IE demand higher character accuracy, it also expects the targeted data to have a predictable order.

ISRI has recently performed IE research that validates these claims. Our current research focuses on the identification of privacy information in free text. The Licensing Support Network (LSN), a large document collection made up mostly of OCR'd documents, should not disclose any personal privacy information once the collection is made publicly available via the Internet. Several "privacy types" must be identified and potentially redacted prior to a document's release. In this section we discuss three of our more well-developed extraction techniques that identify personal addresses, birth dates, and employee identification numbers, and discuss the effect of OCR on the results.

Our address extraction task applies the Hidden Markov Model (HMM) for discovering non-commercial addresses in free text[5, 7]. With the OCR text, we encountered several difficulties. The order of the address components in the OCR output did not always correspond to the correct order of the address on the image. Incorrect zoning affected the tagging applied by the HMM because the order of the tags did not fit the algorithm's expected sequence. Misrecognized characters and the loss of contextual information also interfered with address identification. Our test results indicate that OCR data significantly degrades tasks such as searching for specific features such as addresses in free text. For a more complete discussion of this research, please refer to [19].

Birth dates and employee identification numbers (employee id) are also protected from disclosure under Exemption 6 of the Freedom of Information Act (FOIA), 5 USC Section 552(b)(6)[9]. These algorithms look for relationships among features since if a feature or the privacy relationship is missing, the data is not considered private[2, 1]. For exam-

ple, a date in isolation is not private nor does it become private if it is identified as a birth date with an identifier such as "born on." A birth date becomes private when it is associated with an individual. The statement *John Doe was born on 5/17/55* would be considered private but all these features, person name, relationship, and date, must exist to be tagged as such. The employee id extractor is similar in spirit in that to be considered private, it must be associated with an individual. So it is clear that certain keywords are required for each PA type to be identified and that the proximity of these features to each other is vital to the extraction algorithm.

We evaluated the results using the standard measures of recall, precision, and the F1 measure but in two ways. First, we computed these measures on a per-document basis. At this level, the effectiveness of our programs as document classifiers is determined. That is, we can determine how effective our extractors are at answering the question, *which document contains at least one example of private information of the given type?*

Second, we also evaluate our programs with respect to how well they identify specific instances of a given PA type. Thus, we can determine how well these techniques identify instances of a given PA type. Figure 1 presents the results of experiments using both the birth date and employee id extraction programs.

Although some degradation is apparent at the document level, it is when one compares the results between the clean and OCR instances that the affect of misrecognition is significant.

As with the address HMM IE algorithm, two types of OCR-errors affected the extractors ability to identify birth dates and employee ids: character recognition errors and zoning errors. We estimated that 37.5% of the errors were caused by misrecognition and 63.5% were due to erroneous zoning.

Since most extraction tasks require identifying specific pieces of information from text and will more likely require some proximity, if any part of that information is incorrect it will negatively impact the extraction algorithm. For extraction tasks, close to 100% OCR accuracy is needed. Some of the character errors can be automatically corrected using an aggressive OCR error correction system like Manicure (See Section 3). But we believe if MCT (Section 5) is configured specifically for a particular application, semi-automated correction can significantly improve OCR text for extraction.

3 Automated Post-Processing

Through experimentation and analysis, we identified certain OCR problems that could be corrected through automated post-processing. Mani-

Clean				
Task	Collection	Recall	Prec.	F1
Per Document				
employee id	empid-617	0.979	0.903	0.940
employee id	empid-579	0.949	0.968	0.958
birth date	dob-745	1.000	0.930	0.964
birth date	dob-076	0.974	0.961	0.967
Per Item				
employee id	empid-617	0.695	0.683	0.689
employee id	empid-579	0.815	0.783	0.799
birth date	dob-745	0.990	0.850	0.915
birth date	dob-1076	0.678	0.779	0.725
OCR				
Task	Collection	Recall	Prec.	F1
Per Document				
employee id	empid-617	0.979	0.903	0.940
employee id	empid-579	0.928	0.968	0.947
birth date	dob-745	0.925	0.925	0.925
birth date	dob-076	0.869	0.957	0.910
Per Instance				
employee id	empid-617	0.644	0.556	0.597
employee id	empid-579	0.712	0.406	0.517
birth date	dob-745	0.938	0.835	0.884
birth date	dob-1076	0.464	0.534	0.497

Figure 1: Comparing clean text to OCR text for data extraction

cure (Markup ANd Image-based Correction Using Rapid Editing) is ISRI’s software solution for performing automated OCR post-processing[18]. Manicure is specifically designed to prepare text collections from printed materials for information access applications. We found that by post-processing the *complete* document, misrecognized words could be corrected and other post-cleanup could be performed with a very high level of accuracy[13, 12]. Manicure automatically detects and corrects OCR spelling errors by using dictionaries, approximation matching, the knowledge of typical OCR errors, and frequency and distribution of words and phrases in a document.

Over the past several years, Manicure’s most applied resource has been its ability to automatically correct misspellings in OCR text. The number of corrections made seemed impressive but there was no measurable proof that Manicure made a significant difference in error correction. In 2003, ISRI prepared an experiment that tested the corrections made by Manicure. These tests showed that the word accuracy improvement of Manicure output over raw OCR output ranged from 0.51% up to 2.55% (depending on the quality of the document) for all non-stopwords. This percentage of improvement is comparable to correcting as many as 30% of the misspellings. In addition, these results

showed that the improvement provided by Manicure increases as page quality decreases[8].

These tests confirmed that Manicure improves the quality of OCR text for retrieval and has the potential to increase average precision. For many conversion projects, a Manicure’d collection may be all that is needed. But Manicure also provides a semi-automated correction interface that can bring a document to any desired level of accuracy. As mentioned in Section 2, some applications require a very high level of accuracy to achieve satisfactory results. The following Sections explain how a document’s accuracy can be automatically determined using Manicure and then describes the Manicure Correction Tool (MCT), an interface that can be used to quickly and efficiently bring a document up to the desired level of accuracy.

4 Measuring Accuracy

Having an automated way of measuring OCR quality is a critical component to any large document conversion project. Yet, the only measures available were the estimated character accuracy provided by the OCR or time-consuming ground-truthing and sampling. ISRI found a solution to measuring OCR quality by building on information we already had. Using word accuracy, actually *non-stopword* accuracy, instead of character accuracy was the first necessary modification to enhancing the quality control (QC) process.

To estimate non-stopword accuracy of recognized documents, statistics about non-stopwords must be collected. For example, the QC procedure must know the number of misspelled and correctly spelled words in a document to calculate its percentage of non-stopword correctness. Since Manicure has access to this data as it processes a document, it made sense to include the QC procedure within Manicure.

The intelligence applied in Manicure QC is actually discovered during the automated correction performed by Manicure.

No other QC system can approximate actual OCR document accuracy in the same way Manicure QC does. Figure 2, the pseudocode for the algorithm, shows the additional information provided to and exploited by Manicure QC which vastly improves its ability to verify word correctness.

Stated simply, the QC process takes the number of correct words and misspellings on each page and for the entire document and computes a ratio. The document’s accuracy is then compared to a predetermined threshold (set for a particular application). This process is completely automated. Documents that do not pass QC can be marked for subsequent review.

The QC procedure included in Manicure is a nec-


```

Given document D,
Assume w1,w2, . . . ,wn is the list of words in D,

Set CorrectSum (C) to 0;
Set MisspellingSum (M) to 0;

For each word w in D do:
  if w is in the dictionary then
    mark w as 'correct';
    increment C;

  else if w is an acronym then
    mark w as 'correct';
    increment C;

  else if w is a proper noun then
    mark w as 'correct';
    increment C;

  else if w is a stopword (e.g., the, and, of)
    mark w as 'reject';

  else if w is one of the known patterns (e.g., email,identifiers)
    mark w as 'reject';

  else
    mark w as 'misspellings';
    increment MisspellingSum;
  end if
end For;

Set Accuracy Level (AL) to C/C+M;

If AL is greater than Threshold (Td)
  D passes QC
else
  D fails QC;

```

Figure 2: Pseudocode for Manicure QC

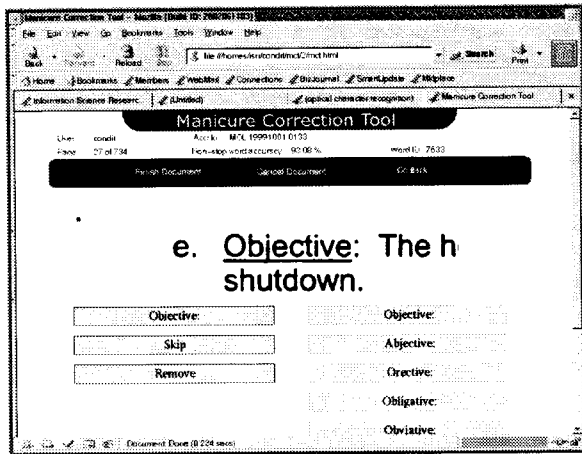


Figure 3: MCT image-based OCR correction

essary prerequisite for documents being submitted to the LSN since based on ISRI research, a document must meet 95% non-stopword accuracy. In some preliminary runs, Manicure QC identified problems with the document conversion process that was causing documents to fail. Further, through verification, we found that if a document passes Manicure QC, it meets the 95% accuracy requirement we believe to be necessary to assure effective retrievability. For a complete description of the Manicure QC evaluation, see [11].

5 The Manicure Correction Tool

MCT is the semi-automated interface that can be used in conjunction with Manicure QC to manually improve documents that fail quality control. The use of MCT would depend on the requirements of a particular application and may or may not be incorporated into the process work flow. But if there are accuracy requirements, MCT provides an easy to use and efficient means for correction.

Documents processed by Manicure are initially scanned to produce electronic images of each page. MCT uses these electronic images in its user interface (See Figure 3). Note that the misspelled word is highlighted on the image in context. Below the fragment of text on the left is the word as it was recognized by the OCR device and on the right is a list of candidate words for correction. A user of MCT need only select the correction from the candidate list and move to the next misspelling. As corrections are made, non-stopword accuracy increases (See counter at top of Figure 3) until the minimum accuracy level is reached.

Behind the candidate list is an elegant algorithm for ordering corrections. It includes dynamic string matching together with a Bayesian technique for selecting OCR confusions. This approach typically produces the most likely candidate at the top of the

list.

First, MCT uses information from Manicure to augment the dictionary (acronyms, proper nouns). In this way, the dictionary becomes dynamic and specific to the document being corrected. Further, MCT applies a priori data and Bayesian techniques to build a “learned confusion matrix,” which is used to order the corrections. The corrections list will then present the most likely candidates based on the knowledge provided to it from Manicure. Figure 4 presents pseudocode in narrative form for the way MCT orders corrections.

6 Conclusion and Future Work

The process of document conversion is more than the generation of ASCII text for a set of input pages. The process must take into account the eventual application for the generated text. As we emphasized throughout the paper, the quality of the application’s effectiveness is heavily dependent on the quality of the text produced. So any large document conversion process must at a minimum have some form of the components discussed in this paper: 1) Method of conversion (scan & OCR) 2) Complete and consistent QC and 3) Automated and/or semi-automated cleanup and correction. Manicure, Manicure QC, and MCT provide this process flow for obtaining the best quality OCR output for any collection with a minimum amount of effort and human intervention.

With our continued research on information access of OCR generated text, we have identified both new areas for research and changes to our systems that would enhance their usability. First, erroneous zoning was a major issue in all our IE experimentation. One open area for research would be the automated correction of incorrect zones. Correcting zoning errors would be a major contribution to information extraction in this domain. Also, Manicure automatically corrects non-stopwords. This of course is crucial for IR but other text elements prove to be important for other IA tasks. Consider the problem of identifying an employee id or a birth date that has been misrecognized. Although Manicure can recognize special strings, it does not attempt to “fix” anything that is not plausibly a word. Fortunately, MCT can be designed to highlight any string in the text. Modifications can be made to MCT to display special strings for verification, including numeric strings.

Our goal is to completely automate the document conversion process. We believe we accomplished this goal for IR. Since other IA tasks may require a very high accuracy rate, the addition of a semi-automated correction system tuned to a specific application is a satisfactory solution for now.

Assume document D has failed with m_1, m_2, \dots, m_k misspellings, number of correct words C, and number of misspellings, M, then,

```
For each misspelling m in D do
  let p1, p2, ..., pt be the list of correct candidates for m;
  re-rank and prune this list using a priori information,
    Bayesian formula, and similarity threshold, Ts;
  sort new p list in order of similarity to m;
  offer new p list to the user for selection with image of m;
  increment CorrectSum, C;
  decrement MisspellingSum, M;

Set Accuracy Level (AL) to C/C+M;
If AL is greater than Threshold (Td) break;
end For;
```

Figure 4: Pseudocode for ordering corrections in MCT

References

- [1] Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*, 2000.
- [2] Sergey Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, EDBT'98*, 1998.
- [3] Jack G. Conrad and Mary Hunter Utt. A system for discovering relationships by feature extraction from text databases. In W. Bruce Croft and C. J. van Rijsbergen, editors, *Proc. 17th Intl. ACM-SIGIR Conf. on Research and Development in Information Retrieval*, pages 260–270. SIGIR, Springer-Verlag, July 1994.
- [4] William B. Frakes and Ricardo Baeza-Yates, editors. *Information Retrieval, Data Structures & Algorithms*. Prentice Hall, Englewood Cliffs, NJ, 1992.
- [5] D. Freitag and A. McCallum. Information extraction with HMMs and shrinkage. In *Proceedings AAAI-99 Workshop Machine Learning and Information Extraction*, 1999.
- [6] D. Harman. *Information Retrieval, Data Structures and Algorithms*. Prentice Hall, Englewood Cliffs, NJ 07632, 1992.
- [7] T.R. Leek. Information extraction using hidden markov models. Master's thesis, UC San Diego, 1997.
- [8] Thomas Nartker, Kazem Taghva, Ron Young, Julie Borsack, and Allen Condit. OCR correction based on document level knowledge. In *Proc. IS&T/SPIE 2003 Intl. Symp. on Electronic Imaging Science and Technology*, volume 5010, pages 103–110, Santa Clara, CA, January 2003.
- [9] Department of Justice. The Freedom of Information Act (FOIA), 5 USC Section 552(b)(6). <http://www.usdoj.gov/oip/exemption6.htm>. Viewed Sept. 28, 2005.
- [10] G. Salton. *Automatic Text Processing*. Addison-Wesley, Reading, MA, 1989.
- [11] ISRI Staff. Measuring and delivering 95% non-stopword document accuracy. Technical Report 2003-04, Information Science Research Institute, University of Nevada, Las Vegas, September 2003.
- [12] 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.
- [13] Kazem Taghva, Julie Borsack, and Allen Condit. An expert system for automatically correcting OCR output. In *Proc. IS&T/SPIE 1994 Intl. Symp. on Electronic Imaging Science and Technology*, pages 270–278, San Jose, CA, February 1994.
- [14] 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.

- [15] 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.
- [16] 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.
- [17] Kazem Taghva, Julie Borsack, Allen Condit, and Padma Inaparthi. The effects of OCR errors on short documents. Technical Report 94-10, Information Science Research Institute, University of Nevada, Las Vegas, February 1995.
- [18] 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.
- [19] Kazem Taghva, Jeffrey Coombs, and Ray Pereda. Address extraction using hidden markov models. In *Proc. IS&T/SPIE 2005 Intl. Symp. on Electronic Imaging Science and Technology*, San Jose, CA, January 2005.
- [20] 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.

Document Processing and Enhancement

A Document Image Enhancement Module: Perspective Warp Correction

Camille Monnier¹, Vitaly Ablavsky², Steve Holden³, and Magnús Snorrason⁴

^{1,3,4}Charles River Analytics Inc.
Cambridge, MA 02138 USA
{cmonnier, sholden, mss}@cra.com

²Computer Science Dept. Boston University
Boston, MA 02215 USA
ablavsky@cs.bu.edu

Abstract

Optical Character Recognition (OCR) is critical to document management workflow in both Army and civilian scenarios. It is also one of its weakest links, having no turnkey or standardized solution. Low quality documents imaged under adverse conditions exhibit severe image artifacts, such as uneven illumination, low contrast, speckle noise, degraded character glyphs (broken, joined, aliased), rotational skew, and page warp (perspective skew and curved baselines). These artifacts catastrophically degrade OCR performance. Processing large volumes of such documents using manual clean-up is prohibitively expensive.

We are currently developing a system named AIDA which pre-processes document images (bitmaps) to boost OCR performance. AIDA consists of a set of document image quality metrics, each detecting a specific artifact type, and a suite of corresponding image enhancement algorithms. AIDA's greatest novelty is a knowledge-based approach to automatically determine the optimal enhancement operator sequence and parameter values, based solely on computed quality metrics. In its current stage of development, AIDA successfully detects and corrects artifacts including rotational page skew, perspective skew, uneven illumination and speckle noise in both English and Arabic documents, and has specifically been validated on Arabic datasets of interest to the Army.

We present the method and results from AIDA's perspective rectification module. Documents captured by hand-held devices such as digital cameras often exhibit perspective warp artifacts. These artifacts pose problems for OCR systems which at best can only handle in-plane rotation. We propose a method for recovering the planar appearance of an input document image by examining the vertical rate of change in scale of features in the document. Our method makes fewer assumptions about the document structure than do previously published algorithms.

1. Introduction

When a document image is captured with a hand-held

device the relative orientation affects the resulting appearance. If the camera's optical axis is co-aligned with the normal of the document plane, such as in a flatbed scanner, the only artifact will be the rigid rotation. This can be corrected using a variety of methods, such as by examining projection profiles [1], Hough transforms [2,3], by line extraction [4], or by nearest-neighbor searches [5], and some modern OCR systems now include this capability as a pre-processing option.

When the optical axis is at an angle with the normal of the document plane, the deformation is no longer simply a rigid rotation, but becomes a perspective distortion that causes variations in text size perpendicular to the axis of rotation.

Methods to correct perspective warping using vanishing point detection have been developed in both architectural photogrammetry [7,8] and in image document analysis [9], most of which involve extracting multiple straight lines that are grouped and intersected to compute a vanishing point. Unfortunately, while it is generally straightforward to extract lines from architectural imagery, it is much more difficult to reliably extract lines from document images. Although it is possible to detect a horizontal vanishing point along the direction of text baselines [9], most methods for rectifying the vertical (baseline-orthogonal) component of a perspective projection are heavily based on assumptions about document structure. Typical approaches for detecting a vertical vanishing point in image document analysis include:

- Assume that a document page has visible left and right boundaries; use those two lines for finding the second vanishing point
- Assume that the text is justified and use the text edge boundaries to define the vertical vanishing point

These methods place rather strong constraints on the subject matter, and do not take full advantage of the rich information available on the page.

The deficiency of the above methods is illustrated in Figure 1 below:

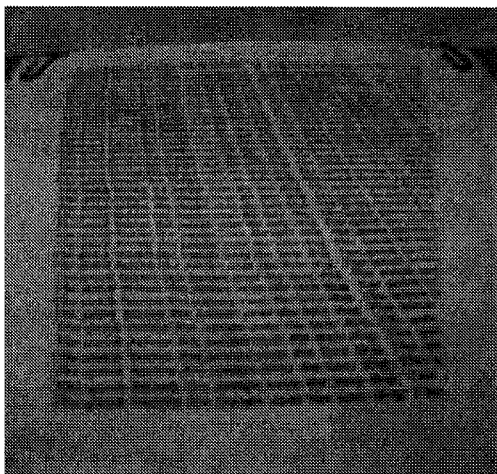


Figure 1: Counter-perspective illusion

The document in Figure 1 has uneven left and right margins and when viewed at an angle appears to have no perspective distortion at all. Although “illusory cues” can be found in this particular instance because each line happens to repeat the same sentence, one can trivially construct examples where such cues would be absent.

2. Overview of the approach

Figure 2 below outlines the algorithm and the subsequent chapters explain the ingredients.

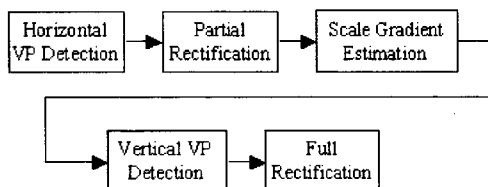


Figure 2: Algorithm description

Text lines are grouped together and their intersection point is used to compute the first vanishing point, which is sufficient for rectification of the horizontal distortion. Next we examine the scale gradient along the direction perpendicular to text baselines to estimate the second vanishing point, which will complete the rectification process.

3. Detection of the Horizontal Vanishing Point from Text Baselines

Rectification of perspective distortion has successfully been accomplished by detecting vanishing points in a scene. Work has been done regarding the 3D reconstruction of structure from imagery containing man-made architectures via the use of vanishing points [7,8]. In the case of document images, where long,

straight edges are rare, it is required to perform additional analysis to extract the visual cues necessary for vanishing point detection. We have implemented a version of Pilu’s algorithm [9] for detecting and grouping text lines in order to compute the horizontal vanishing point and rectify the perspective distortion along text lines (Figure 3).

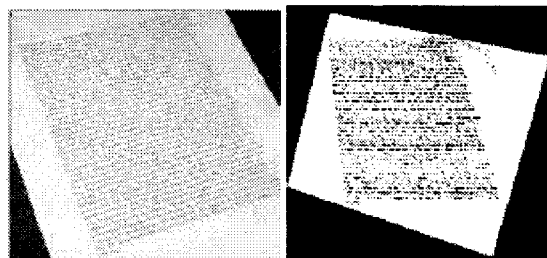


Figure 3: Rectification with respect to the first vanishing point

4. Perspective Rectification from a Vanishing Point

Given the location of a vanishing point $\mathbf{V} = \begin{bmatrix} V_x \\ V_y \end{bmatrix}$, it

is possible to write a planar homography that corrects the deformation produced by a perspective projection up to an aspect ratio [7,9]. In the case of a document image, we expect to detect at most two vanishing points in the directions parallel or perpendicular to text baselines. We compute the homography for correcting the distortion in the direction of the text lines as in [9]. Pilu writes the homography as a rotation to align the vanishing point with the image X-axis followed by a transform \mathbf{M}_x that places the detected vanishing point at infinity along the X-axis (1).

$$\mathbf{M}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{V_x} & 0 & 1 \end{bmatrix}, \quad \mathbf{M}_y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{1}{V_y} & 1 \end{bmatrix} \quad (1)$$

We compute the homography for the vanishing point perpendicular to the text lines as a translation along the X-axis to set the vanishing point’s X coordinate as the origin, followed by a transform \mathbf{M}_y that places the detected vanishing point at infinity along the Y-axis (1).

5. Feature Extraction

We assume that text lines may be non-justified, and that line endpoints are thus unreliable for computing the vanishing point. We instead rely on text baselines as our primary source of data, choosing as features the average inter-line distance and average line height (scale), and the rate of change of scale over the image surface (texture gradient).

document image, $\gamma = 0.0195$

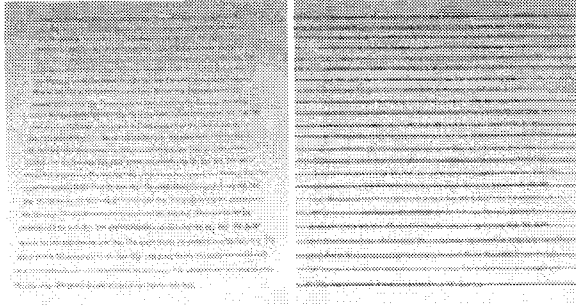


Figure 4: Camera image of a slanted ($\approx 42^\circ$) text document (left), and recovered text baselines (right).

5.1. Baseline Detection

We estimated baselines by implementing a variation on Pilu's method [9] for detecting text lines in a document image. The image is binarized and processed for connected components, which are then grouped using a line-growing algorithm (Figure 4). While this is not necessarily the most robust baseline detection method available, it is useful because it extracts the features we need for detecting both the horizontal and vertical vanishing points.

5.2. Line Spacing Gradient Estimation

We estimate the gradient of inter-line spacing as a linear function over image pixels along the y image axis. If a horizontal vanishing point is detected, the correction is applied prior to estimating this metric. In order to reduce noise, we fit the y -locations of detected baselines to a linear function of the form $\Delta y' = ay' + b$; this mitigates the impact of outliers on the final estimation, and helps smooth noisy data points (Figure 5). Note that any known uniform distance in the document (such as text height) may be used to estimate the gradient, though larger features tend to provide much more stable estimates.

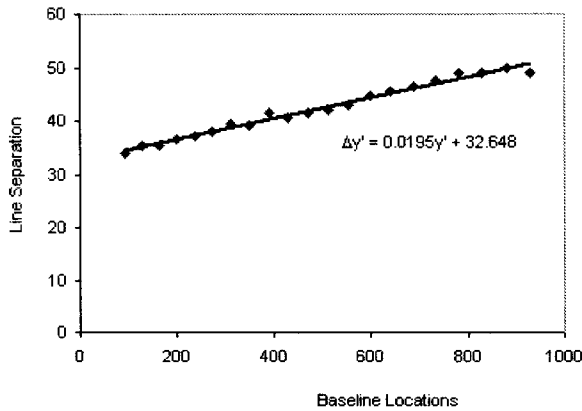


Figure 5: Inter-baseline distances of the above

6. Vertical Vanishing Point Estimation

We approach the problem in terms of the measurable quantities described in Section 5. Our solution assumes that text lines are for the most part evenly spaced in the original document, and that the document image is approximately centered about the camera's optical axis.

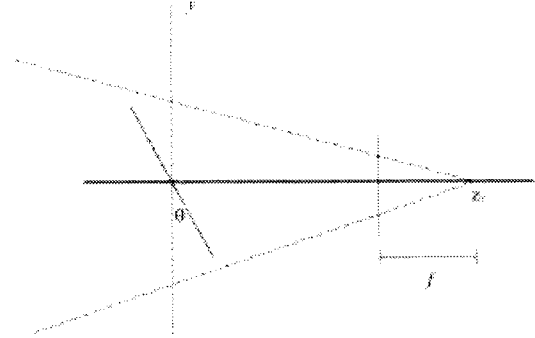


Figure 6: Geometry of the perspective projection of a rotated plane

Treating the deformation as the perspective transformation of a page in the XY plane that has been rotated an angle θ about the X -axis (Figure 6), and discarding the Z component, we obtain the 2D deformation:

$$\begin{bmatrix} x' \\ y' \\ h \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & 0 \\ 0 & \frac{-\sin\theta}{f} & \frac{z_c}{f} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2)$$

Where z_c is the camera's position along the Z -axis and f represents the distance from the camera to the image plane. Writing y in terms of discrete lines, with line number n and uniform spacing d , we obtain

$$y = nd \quad (3)$$

which we can substitute into (2) to generate the relationship

$$y' = \frac{fnd \cos\theta}{(z_c - nd \sin\theta)} \quad (4)$$

The vanishing point along the Y -axis can be defined as the mapping of a point at infinity to a finite point in the image plane. Taking the limit of (4) as n goes to infinity, we obtain the equation for the vanishing point

$$y'_{vp} = \lim_{n \rightarrow \infty} y' = -f \cot \theta \quad (5)$$

We wish to describe the vanishing point in terms of measurable quantities $\Delta y' = \frac{dy'}{dn}$ (the separation distance between distorted lines) and $\gamma = \frac{d(\Delta y')}{dy'}$ (the rate of change of the separation distance between distorted lines). Setting

$$\Delta y' = \frac{dy'}{dn} = \frac{fdz_c \cos \theta}{(z_c - nd \sin \theta)^2} \quad (6)$$

and

$$\frac{d(\Delta y')}{dn} = \frac{2fd^2 z_c \sin \theta \cos \theta}{(z_c - nd \sin \theta)^3} \quad (7)$$

Combining (6) and (7) we can write

$$\begin{aligned} \gamma &= \frac{d(\Delta y')}{dy'} = \frac{d(\Delta y')}{dn} \cdot \frac{dn}{dy'} \\ &= \frac{2d \sin \theta}{(z_c - nd \sin \theta)} \end{aligned} \quad (8)$$

We can then rewrite equation (6) in terms of our measurable quantities $\Delta y'$ and γ by substituting equations (6) and (8) into an expression for $\cot \theta$:

$$\cot \theta = \frac{2\Delta y'(z_c - nd \sin \theta)}{f\gamma z_c} \quad (9)$$

Substituting into (5), we have

$$y'_{vp} = \frac{2\Delta y'(z_c - nd \sin \theta)}{\gamma z_c} \quad (10)$$

Because we assume that the document image is approximately centered about the optical axis, it is reasonable to state

$$z_c \gg nd \sin \theta \quad (11)$$

for points near the origin. We thus obtain our estimate for the vanishing point,

$$y'_{vp} \approx \frac{2\Delta y'}{\gamma} \quad (12)$$

where $\Delta y'$ is the scale value near the origin, and γ is the rate of change in scale in the image. Applying the vanishing-point correction, we obtain the rectified image (Figure 7). Note that the vanishing point is given with respect to the origin, which is assumed to be near the center of the image. For documents that have a significant horizontal skew (Figure 9), it is necessary to

estimate the vanishing point's horizontal offset. We are able to estimate the vanishing point's horizontal offset using a single vertical cue, such as left-most baseline endpoints. While we avoid searching for vertical cues in the computation of the vanishing point's Y-component, we only require a single vertical cue to determine the corresponding X-coordinate.

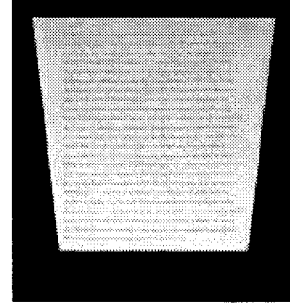


Figure 7: Rectified document image

7. Results

Our method for the rectification of perspective-skewed document images works well over a large range of slant angles (Figure 8), and requires only a single vertical cue to successfully rectify arbitrary perspective distortions (Figure 9). Our preliminary results indicate that a significant boost in OCR performance can be expected for documents rectified in this manner (Table 1). While very little difference in performance was observed at small skew angles, at larger angles OCR performance increased by as much as 5% in our trials. In the sample set shown below, a slight loss in performance (0.2%) is observed in the small-angle case due to blurring caused by the transformation. Improvement in OCR performance across various document images is directly related to the proportion of text that lies furthest from the camera (and is thus the most affected by the perspective skew). Consequently, at skew angles exceeding 35°, we notice an attenuation of OCR improvement as the effective resolution of the text furthest from the camera is often too low for warp correction alone to aid recognition. In practice, our OCR did not exhibit difficulty in recognizing text closest to the camera in all but the most severely skewed images.

Param.	Original	Rectified
$\theta = 10^\circ$ $\gamma = .0067$ $\Delta y' = 50.6$		
$\theta = 22^\circ$ $\gamma = .0127$ $\Delta y' = 45.6$		
$\theta = 34^\circ$ $\gamma = .0171$ $\Delta y' = 40.7$		

Figure 8: Secondary rectification of document images slanted about the X-axis

Table 1: Sakhr Automatic Reader's performance before and after rectification

Skew Angle	OCR Performance (Skewed)	OCR Performance (Rectified)
10°	93.3	93.1
22°	85.4	90.9
34°	81.0	85.9

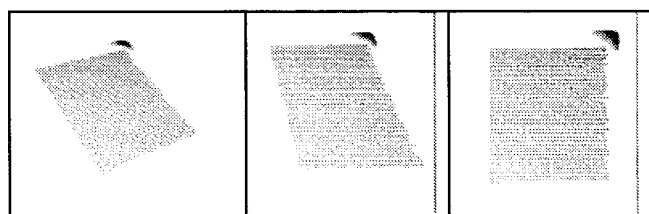


Figure 9: Binarized document image taken from an arbitrary perspective (left), partially rectified

document image (middle), fully rectified image (right)

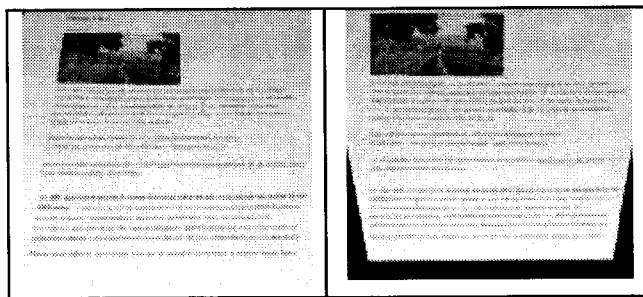


Figure 10: Document image at 60° (left), rectified image (right)

8. Conclusions

We present a method for rectifying perspective distortion in the direction perpendicular to text lines in document images. Provided the original document has a reasonably well-distributed number of evenly spaced or uniformly tall text lines (Figure 10), our algorithm is capable of reliably correcting the distortion with no additional input. In cases where the image's center does not coincide with the optical axis, a vertical text cue must be detected to determine the vanishing point's horizontal offset. While it is reasonable to assume that most text is left-justified (right-justified in the case of Arabic), we are pursuing a method to more robustly estimate a scale gradient along the X-axis. Additionally, while our method is numerically stable with respect to noise, it is likely that its performance can be further improved by exploiting an alternate estimate of the texture gradient such as scale-space analysis [6].

9. References

- [1] A. Bagdanov and J. Kanai, Projection Profile Based Skew Estimation Algorithm for JBIG Compressed Images, *Proceedings of the 4th International Conference on Document Analysis and Recognition*, Ulm, Germany, 1997, pp. 401-405.
- [2] H.-F. Jiang, C.-C. Han, K.-C. Fan, A Fast Approach to the Detection and Correction of Skew Documents, *Pattern Recognition Letters*, 18 (7), 1997, pp. 675-686
- [3] Amin, A., Fischer, S., 2000. A document skew detection method using the Hough transform. *Pattern Analysis and Applications* 3 (3), 243-253.
- [4] Cao, Y., Wang, S. Li, H., Skew detection and correction in document images based on straight-line fitting, *Pattern Recognition Letters* 24 (12), 2003, pp. 1871-1879
- [5] Lue, Y., Tan, C.L., A nearest-neighbor chain based approach to skew estimation in document images, *Pattern Recognition Letters*, 24 (14), 2003, pp. 2315-2323
- [6] Lindeberg, T. "Scale-Space Theory: a Basic Tool for

Analyzing Structures at Different Scales," *Journal of Applied Statistics*, vol. 21, no. 2, pp. 225-270, 1994.

- [7] Liebowitz D, Criminisi A, Zisserman A. Creating architectural models from images. In *Proc. Eurographics*, Milan, Italy, Sept. 1999, pp.39-50.
- [8] F.A. van den Heuvel, Vanishing point detection for architectural photogrammetry, *Int. Arch. Photogrammetry Rem. Sens.* 32 (part 5) (1998) 652-659.
- [9] Pilu, M. "Extraction of Illusory Linear Cues in Perspectively Skewed Documents," *CVPR 2001, also a Hewlett-Packard UK tech report*.

The research reported in this document/presentation was performed in connection with contract/instrument DAAD17-03-C-0009 with the U.S. Army Research Laboratory. The views and conclusions contained in this document/presentation are those of the authors and should not be interpreted as presenting the official policies or position, either expressed or implied, of the U.S. Army Research Laboratory or the U.S. Government unless so designated by other authorized documents. Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

Document Recognition and Translation with Handheld Mobile Devices

Esin Darici Haritaoglu Ismail Haritaoglu

Polar Rain Inc.

465 Fairchild Drive Suite: 226

Mountain View, CA 94043

Abstract

We describe a mobile system which is capable of taking a picture of a document or a sign (written in foreign language) with digital camera attached to commercial off the shelf (COTS) mobile digital personal assistant (PDA) and process the captured image to extract foreign text from images, recognize text (Latin and Arabic alphabet), and translate the document into English so that Soldiers can understand what is written on these documents or information signs without requiring a wireless connection. We have achieved more than 98% accuracy for both scripted Arabic and Latin OCR engines and overall process including detection, extraction, recognition and translation takes less than 15 second for a half-page document written with 12 point fonts.

1 Introduction

In this paper, we would like to introduce our mobile document recognition and translation system and challenges we have faced and solutions we have provided during development of our system¹ Our system consists of a PDA device and attached digital camera. All processing modules including Latin or Arabic OCR and multilanguage translation engines are working on PDA and do not require any wireless communication to complete the process. Our system is capable of providing OCR and translation results less than 15 second for half page documents. Example of sign translation is shown in Figure 1.

One of the most challenging parts was that all solutions for OCR and translation should fit into the small memory space and limited processing power of mobile

devices. Additionally, the image quality of available PDA based digital cameras and limited camera operation (manual focus, handheld motion while capturing images) makes the problem more challenging. We will explain our mobile Arabic and Latin OCR and translation algorithm which yield very accurate OCR and domain specific translation results.

Soldiers engaged in battlefield operations in foreign countries encounter a variety of foreign language documents and signs of unknown content. These documents and signs might be relevant to the Soldier's current mission or to the overall battlefield process. Currently US Department of Defense and Security agencies interest is to use such mobile translation device in foreign countries to translate documents and road/warning signs so that Soldiers are able to read road/information signs and documents and quickly triage and translate foreign documents so that mission relevant documents and information can be utilized in a timely manner.

Our system permits the user to translate captured foreign documents quickly so that mission relevant documents can be utilized in a timely manner, and documents containing information relevant to commanders on the battlefield can be passed to the appropriate evaluation personnel. The overall concept is to prevent documents containing time-critical information from being combined with large quantities of other documents that will be stored for future processing.

In addition to a document translation system, our prototype device will be capable of permitting users to capture road/warning/information sign and documents and translate to English quickly so that users can read or interpret foreign sign containing warning, navigation or other information relevant to mission. The translation tool will help user for accurate decision making by providing translation when and where they need most.

Core elements of our technology are the ability to extract text information accurately and recognize foreign text quickly with a limited resource mobile device. We have developed very fast and accurate image processing algorithm to be able to extract text information reliably

¹ The research reported in this document/presentation was performed in connection with contract/instrument W911QX-04-C-0015 with the U.S. Army Research Laboratory. The views and conclusions contained in this document/presentation are those of the authors and should not be interpreted as presenting the official policies or position, either expressed or implied, of the U.S. Army Research Laboratory or the U.S. Government unless so designated by other authorized documents. Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

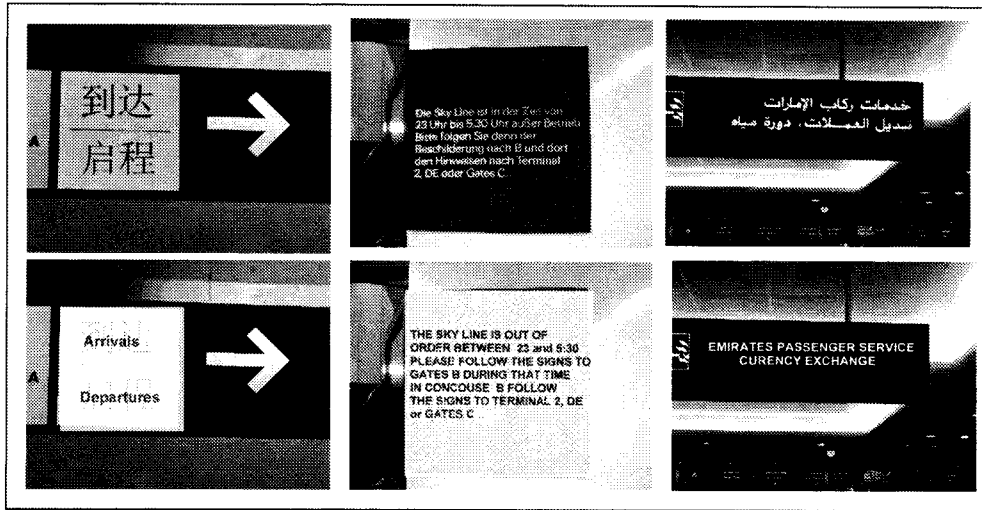


Figure 1: Example of sign translation with mobile devices to help navigation

from pictures and documents and recognize each character or word in the text. It does not require any special hardware and works on a COTS PDA.

Once foreign text detected and characters and words are extracted, they are recognized using Polar Rain's mobile OCR engine. Polar Rain's Mobile OCR engine is capable of recognizing machine printed Latin, Arabic and Cyrillic documents and road/information signs. It is capable of generating acceptable and usable OCR results as small as 125 dpi document captured with PDA's digital cameras. Once the text recognized it is translated to English. Polar Rain has been developing custom domain-specific and generic mobile translation engines which is capable of translating Spanish, Arabic, Serbian to English.

All operation is working on PDA with small digital camera and Soldiers are able to get translation ready in less than 15 second. All processing is done in mobile device without requiring an wireless connection. End users can use the device for any road/sign information translation as simple as taking a picture for other purposes, such as traveling and navigation in foreign in foreign countries

In this paper, we will first introduce our system hardware and some of the limitations in Section 2. Then in Section 3, we will go over our Latin and Arabic OCR engines that we have developed and their accuracy for test databases with different size text. In section 3, we will explain mobile translation and triage engines that we are using in the system and performance analysis for different sets of documents and road images.

2 Mobile Translation and Triage System without Wireless Connection

Our goal was to develop an image based document/text translation and triage system using mobile device with which users only need to capture written document or information/or road sign and our system provides transla-

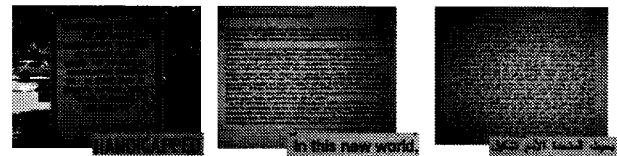


Figure 2: Example of images captured with VEO 1.3MP PDA camera

tion of written text in those images. Although image capture and text detection, OCR and translation technologies has been investigated for at least 25 years, the challenging problem is to make the system running on a small mobile device which has many restrictions such as limited main memory, and limited processor power. This makes the problem harder and selection of algorithm is crucial to obtain acceptable performance, e.g., complete the processing time in an acceptable amount of time with acceptable OCR and translation accuracy. In addition, we would like to implement the system without any wireless connection. If the system has a wireless connection the images can be transferred to host computer where image processing and translation are applied and results are returned to user over wireless link. Although this approach is good for both commercial and military application [9, 10, 8], the wireless link may not be available all the time on operation fields therefore those systems may not be useful if there is no wireless connection.

We have implemented our system for Commercial of the shelf (COTS) Personal Digital Assistant (PDA) which has attached digital camera. Although there are sufficient alternatives for PDA's on the market, there is only one or two good quality digital camera which can be attached and used with PDA's. In our system we have chosen VEO 1.3MegaPixel SD digital camera with manual focus. We can capture up to 1280x1024 pixel resolution. Although there is some color quality problem, it seems that VEO 1.3 SD camera gives the best per-

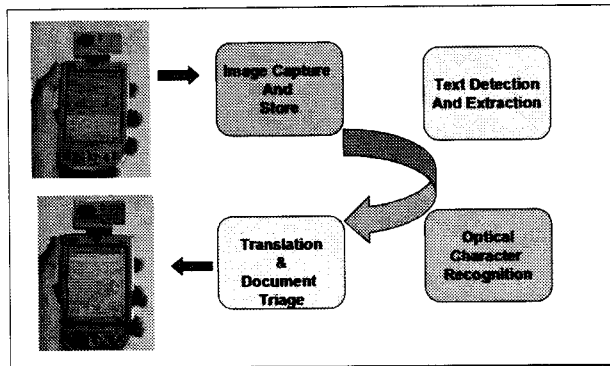


Figure 3: Mobile Document Translation Device and Modules

formance in our application over other alternatives. Recently, some of the PDAs has built-in camera on board but none of them has ability to focus document images in short distance. VEO allows us to adjust the focus so that we can capture document/pages in short distance (as small as 8 inches) so that we can capture quarter of a page in sharp focus. In Figure 2, an example of document and road sign images taken with VEO are shown.

Figure 3 shows overall system diagram. It has four main modules:

- Image Capture and Store** Our system allows users to capture images with 1280x1024 resolution. Inherently, motion blur and de-focus are two important problem while capturing image with VEO or other mobile digital cameras. Our system has image enhancement modules to determine the amount of blur and de-blur some parts if necessary.
- Text Extraction and Layout Analysis** Our system has two main text detection and extraction modules for road/information signs and document pictures. Road sign detection modules allow us to detect text on road and information signs which has various color of the text, orientation and illumination. Road sign detection and extraction modules use sophisticated algorithms to detect scene text which is inherently difficult to detect. Our system uses very fast text detection and extraction modules for document images. Document images has more text and less non-text regions and detection and extraction of text does not required sophisticated image enhancement to extract the text. Our system has text detection and extraction for both Arabic and Latin text which uses different approaches to extract the each character as Arabic text has scripted nature which is different than Latin text.
- Character Segmentation and Recognition** We have developed two different OCR for mobile devices: Latin and Arabic OCR. Both OCR uses shape similarities, size and font sensitive information while recognizing character. They also use

some language depended features to increase the accuracy. We have tested both OCR engines using both road signs and document images and we have obtained more than 98% OCR accuracy for Latin and Arabic text.

- Text Translation and Triage** We have developed multi language translation from Spanish, Arabic and Serbian to English. In addition we have developed mission sensitive dictionary which allows us to triage documents/text quickly.

3 Text Detection and Extraction for Document and Road Signs

As text detection and extraction in document images is slightly less complicated than text detection and extraction in road sign images, we have developed two different approaches to detect text in road sign and document images. Main motivation is that road/information images have less text but complicated background difficult to detect, however document images has less difficult background but more text. Therefore we would like to allocate more time to detect text in road/information sign processing but less time for recognition. In contrast, more time for recognition and less time for detection in document images.

We have developed very robust but computationally expensive text detection and extraction for road sign images. We have applied this method to text images without any major changes and we have obtained good results. However we have noticed that we can use less expensive vision techniques to obtain the similar results on document images in less amount of execution time. For example, the assumption we can make is that the document images are not as complicated as road signs and there is unique background color (such as white) and unique foreground color (such as black). In addition, the quality of the VEO images makes the text extraction little easier. We have developed two different text detection and extraction algorithms (a) Road Sign Text Detection (RSTD) (b) Document Text detection (DTD). As road sign images may have very complex background and variation, we prefer image enhancement before text detection and segmentation. We apply the hierarchical connected components algorithm to obtain segmentation for road sign images. Symmetrical Neighbor Filter (SNF) enhancement was one of our innovation for road sign detection which we implemented in a fast way to achieve both memory and processing efficient version. It helps to smooth the images by removing the interior region noise while sharpening the edges between the complex background and text regions. Figure 4 shows an example of road sign detection and recognition obtained with our system

However we do not need such complicated algorithm for text documents. We have developed a fast appear-

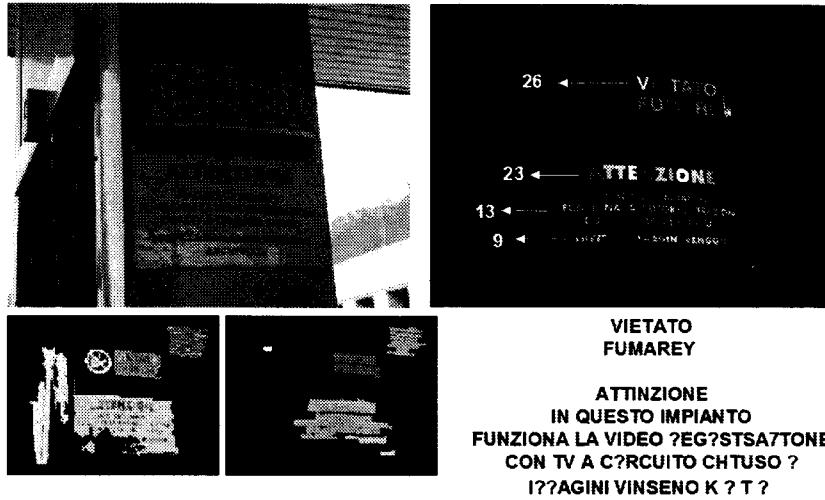


Figure 4: Example of text detection and recognition in road sign images

ance based method which is applied to entire image and utilize local contrast between text and background. Instead of single global thresholding method (which tends to fails when there is camera noise or imager problem), we compute a threshold locally using text appearances. This method yields very good detection and extraction method for document images but does not perform well for road sign detection. Figure 5 shows the performances of both detection method on a PDA with 624 Mhz xScale processor.

4 Optical Character Recognition on Mobile Device

We have implemented shape based optical character recognition for Latin and Arabic text. In order to use both memory and processor power efficiently, we have developed an adaptive feature based recognition algorithm based on character shape invariant and shape depended features. One of the challenging part of the OCR is to develop such recognition system on limited memory and processor power.

4.1 Latin OCR

we have implemented 2D normalized shape template matching method. The basic idea behind that method is to defer the computationally expensive feature comparison as much as possible. We first use computationally fast features to eliminate the non-similar characters and then apply computationally expensive features only to those characters that pass the first level.

Feature selection is critical to obtain a good recognition rate. However, as we are targeting to implement it for PDA, memory usage and speed are other important factors. While the selected features should give the most distinguishable power for all the character that we need to recognize, the recognition and comparison algorithm should be fast enough to produce the results in limited

time and the feature space should not require too much memory. After conducting extensive experiments, we selected 2D-shape histograms which describe the relative orientation and distance between the points in the characters, as main feature for recognition engines. We trained the system using an adaptive training method. We used scene-characters that are detected in image databases in training sets. If there is not enough number of characters, we combined computer generated-characters (where the documents are printed and captured by the camera instead of captured on road sign). We trained 53 characters using more than 6250 single character with different size. Each characters in the training set has 45-55 sample using different fonts and different size (8-72 pixel in height).

We have conducted many test and training session to understand the performance of the recognition system for various Latin characters which is written in different fonts and different sizes. We have used four main fonts (Arial, Verdone, Helvetica, Tohoma) and character sizes (10 point to 56 point) while we are generating characters. We have printed text documents in a one page and captured the entire page with a single image with 640x480 and 1280x1024. This setting provided us almost 75dpi resolution for 640x480 image and 150dpi resolution for 1280x1200 resolution. As we are processing pixel (not the points), we use number of pixel as size for the characters. For example a character printed with 12,16,24,32 point font size appears in document images with 6,9,14,18 pixel in height size for 640x480 document images, 14,20,30,38 pixel in height size for 1280x1200 images. Generally, our recognition system works good for characters which has pixel size 15 or higher.

4.1.1 Latin Character Recognition Performance Evaluation

We have generated three sets of character data for training and testing (a) characters extracted from docu-

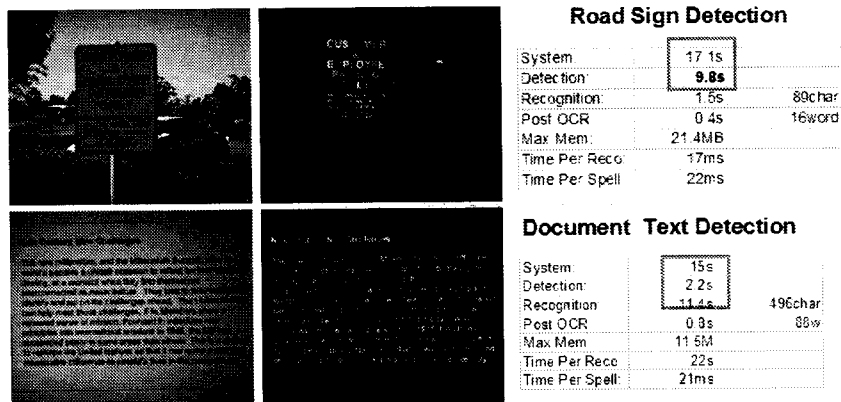


Figure 5: Text detection and extraction performance on a PDA which has 624 Mhz intel Xscale Processor

ment images which is generated randomly and printed 10-18 font size with four different fonts (b) characters extracted from road sign images (used in highways and outside information) (c) characters extracted from document images which contains English text and printed with 10 or 12 points single font. The first set has almost 7600 characters (four different font type with different apparent sizes changing from 8 pixel to 85 pixel in height). The second set has more than 8000 characters and written in the sign with different fonts (we have observed that the font variation is not significant as most of the road has consistent font type). A third set consists of 16235 characters and contains single font. An example of images are shown in Figure 6.

Below are the some character recognition results we have obtained. On the average we have obtained 98.7% recognition accuracy for document images. We have obtained slightly lower recognition rate 97.8% for road sign images. We observed that characters extracted from road sign images have more variation than characters extracted from document images. We were able to obtain recognition rate of almost 99.2% for character Set-II which has apparent size smaller than 19 pixel

	All Sizes	($s \leq 16$)	($s \leq 30$)	($31 \leq s$)
Set I (CG)	0.982	0.929	0.980	0.995
Set II (RS)	0.968	0.968	0.966	0.971
Set III (UN)	0.984	0.991	0.993	0.974

Table 1: OCR accuracy of our system for Latin alphabet

As it is seen above, the best recognition performance is obtained with 10-12 point size printed UN document images. Extracted characters have small appearance size however they only contain single font type.

Overall we have reduced the detection processing down to 1.7 second on the average using 624Mhz xScale based PDA with Intel Compiler. As you can see from the results, the detection process is almost fixed and does not depend on how many characters are extracted from the image. However, recognition process is directly related to how many characters are extracted from the image. For every character extracted by detection process, recognition function is executed, therefore the more character the image contains the longer the overall

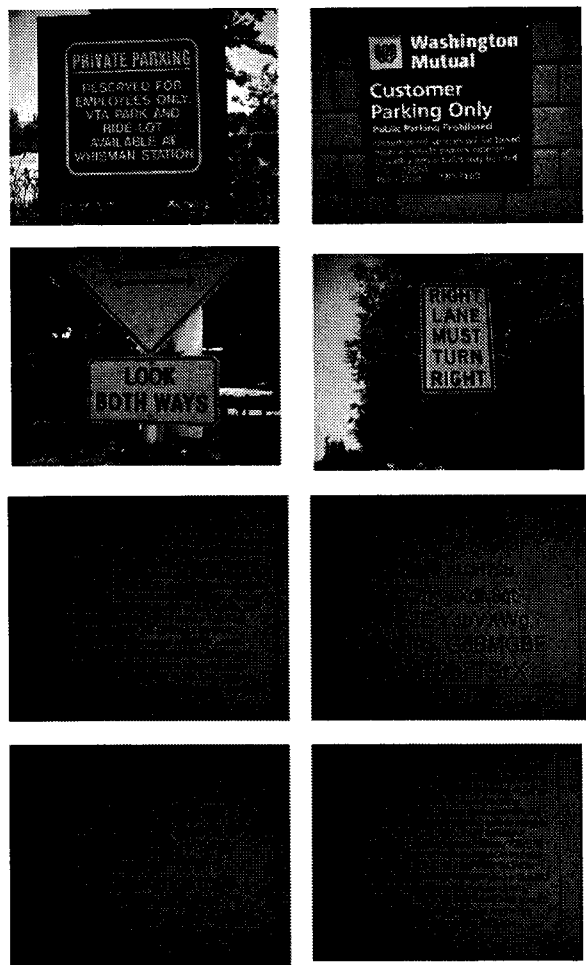


Figure 6: The system has been tested by using two set of image database from which each character is extracted (a) sample road sign images (b) sample document images

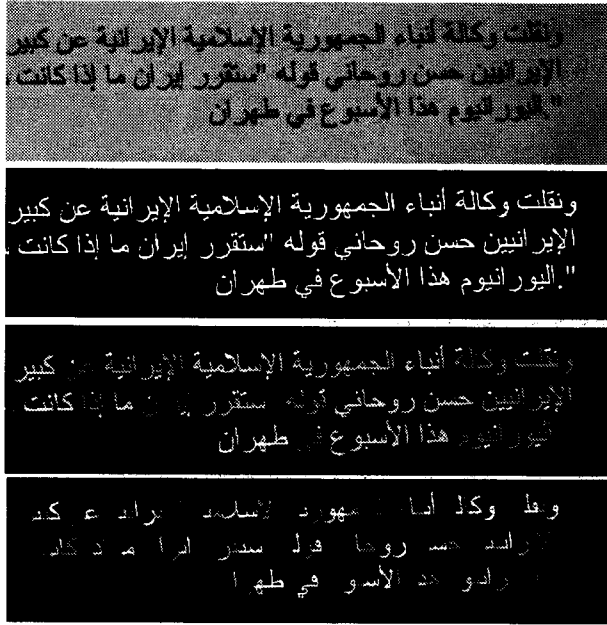


Figure 7: An example of Arabic character recognition process: original text image (top row), detected text only regions (second row), word segmentation (thirds row) and final Arabic character segmentation (last row)

recognition time. In other words, the recognition-time is shorter for images which contain bigger apparent size character than the images which contain smaller apparent size characters. One of the important speed measures is processing time per single character recognition. With our method, we are able to recognize single character in 8 ms on the average.

4.2 Arabic OCR

In addition to Latin OCR, we have developed fast and accurate Arabic OCR which is able to recognize scripted Arabic characters and words. In summary, we first detect the Arabic text in document images and extract the Arabic text from images. Then we apply connected components algorithm to find each connected components. Each connected component may contain single or multiple Arabic characters or a sub-piece of an Arabic characters. After we obtain the connected components, we segment words. After the words are detected we further apply Arabic character segmentation algorithm to detect each individual Arabic character. Example of Arabic character recognition and dot integration are shown in Figure 7.

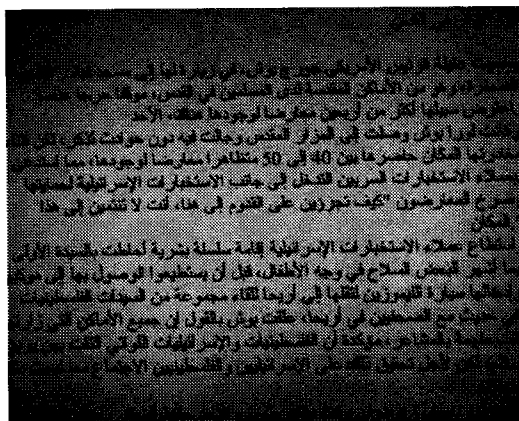
There are mainly 26 Arabic characters and 10 Arabic digits in Arabic alphabet. However, because of scripted nature of Arabic language, each character can have 4 different character shape depending on it's position (isolated, middle, initial or end character). In addition, some of the basic Arabic characters can take some diacritics, dots etc. These dots make the character totally different character. Therefore Arabic characters without dots may

have many ambiguities in recognition. In order to build a robust recognition system for Arabic, we have develop Arabic characters with and without dot information. We trained our system to recognize Arabic OCR characters. Each Arabic OCR character may represent a single Arabic text Character or multiple Arabic text characters depending on the position and whether or not it has dots (Arabic Text characters are the ones used in Arabic text). In our recognition system, we provide OCR-Character to Text character mapping, and if there are one-to-many type of mapping, dot classification algorithm helps to resolve disambiguates.

4.2.1 Performance Evaluation

We have conducted extensive tests to evaluate the Arabic character segmentation and recognition accuracy. We have generated four different test sets, each test set have word-level ground truth data that allows us to compare the results. We have computed three accuracy metrics: character recognition accuracy(OCR), word recognition accuracy without any spelling correction (WORD-NSP), word recognition with Spelling(WORD-SP). Here is the summary of the results:

- **set-I:** Set-I contains 21 document images taken with Veo camera with 1280x1024 resolution. The Arabic text are CNN news segments, UN documents etc and printed on 12 and 16 point size Arabic fonts (roughly equal to 150dpi). It contains 4049 scripted Arabic words and 19210 characters. The words and characters appear on document images with various apparent sizes, where avg. character size is 17 pixel.
- **set-II:** Set-II contains 13 document images taken with Veo camera with 1280x1024 resolution. The Arabic text are CNN news segments and printed on 16 point size Arabic fonts and cover half page document(roughly equal to 175dpi). It contains 1927 scripted Arabic words and 9043 characters. The words and characters appear on document images with various apparent sizes, where avg. character size is 18 pixel.
- **set-III:** Set-III contains 11 document images taken with Veo camera with 1280x1024 resolution. The Arabic text are CNN news segments and printed on 12 point size Arabic fonts and cover half page document(roughly equal to 145dpi). It contains 1735 scripted Arabic words and 8227 characters. The words and characters appear on document images with various apparent sizes, where avg. character size is 14 pixel.
- **set-IV:** Set-IV contains 31 document images obtained with scanner at 300 dpi resolution. This set allow us to compare our results with commercial Arabic OCR products. The Arabic text are CNN news and UN documents and printed on 12 and 16



Kenai: OCR

نورا بوش في القدس واجهت عقبة الرئيس الأمريكي جورج بوش في زيارة لها إلى مسجد قبة القدس الصخرة وهو من الأماكن المقدسة لدى المسلمين في القدس موقفاً حرجياً عندما اعترض سبيلها أكثر من أربعين معارضاً نوجدها هناك الأحد وكثرت نورا بوش وصلت إلى المزار المقدس وجالت فيه دون حوادث يذكر لكن أثناء مغادرتها المكان حاصرها بين لا إلى حال متظاهرا معارضاً نوجدها فيها بعلاء الاستخبارات السريين التمدخل إلى جانب الاستخبارات السرائيلية لجمابها وصرخ المعارضون كيف تجرؤين على القدوم إلى هنا ان ال تتمنين إلى هذا المكان واستطاع عمال الاستخبارات السرائيلية إقامة سلسلة بشرية أحاطت بالسيده الأولى ما شهر البعض السطح في وجه الأطفال قبل أن يستطيعوا الوصول بها إلى موكبا وانخالتها سيرة التيموزين لنقلها إلى أريحا لقاء مجموعة من السيدات الفلسطينيات في حديث مع الصحفيين في أريحا خلقت بوش بانقول إن جميع الأماكن التي زار تن مشهمة بالمشاعر مؤكدة ان الفلسطينيين والسرائيليات اللواتي التقت بهن برين لقاء تكن الرجل تحقيق ذلك على السرائيليين والفلسطينيين اجتماع معا لبحث نر وتحقيقه كانت نورا بوش زارت في وقت مبكر من يوم الحد حان المبكى احد الأمالين

Dict: CNN (22K) in second

System:	10.2
Detection:	0.7
Recognition:	6.4
Spelling:	2.3
Translation:	0.5
Triage:	0.3
# of Char:	846
# of Word:	174
# of Spelling:	43
# of Translation:	198

9.2 MB Memory

Kenai-P: Translation

Laura bush at jerusalem faced [Eqytp] us president bush in visit its to mosque bloc jerusalem dome he is of holy places at muslims at jerusalem stand critical when objected recommendations from more than one forty course such there minimum had Laura bush amounted to [AlmzAr] the [wJAH] its without accidents the the mentioned but during favours place [HARhA] between not to elevated [mZAhRA] course such thus summoned [bEmAlAI] intelligence secret of to mention intelligence [A-lsrAnylyp] accomplishment voice oppositionists how [tjr&yn] on come to here the the [ntmyn] to such a place managed [EmAlAI] intelligence [A-lsrAnylyp] set series human advised sayeda first the month some bek in they [AAITAI] by that able reach by to [mwkbA] kholi cars [Allymwzyn lqthA] to jericho meet a package of women participating in an interview with preas in jericho was completed suspended bush saying that all places [Alty] visited would [mihmp bAlm\$AEr] they affirmed that that behaviour participating [wA-lsrAnylyAt] forced al victim wear al-alam but man achieve a on [A-lsrAnylyyn] palestinians meeting together discuss the warned [wthlqyh] had parker bush visited passed time early of on minimum time yard targets places

Figure 8: Example of results obtained with Polar Rain's Kenai-II Mobile Arabic to English Translation and Triage system. The execution times are in second and obtained with a PDA which has 64MB main memory and 624Mhz xScale processor

point size Arabic fonts. It contains 7695 scripted Arabic words and 34477 characters. Average character size is 25 pixels.

Using above test sets, we have achieved following ocr and word accuracy:

	OCR.	WORD-NSP	WORD-SP
Set I	0.989	0.906	0.946
Set II (12)	0.972	0.775	0.893
Set III (16)	0.929	0.544	0.757
Set IV (scan)	0.991	0.965	0.977

Table 2: Arabic OCR accuracy of our system

Our Arabic OCR is also very fast and runs on a 624 Mhz PDA with 6.2 second per image (per half page) which is rough 7.5 ms per character. Figure 8 and Figure 9 shows examples of Arabic and Spanish OCR and translations results with execution time statistics.

5 Document Translation and Triage

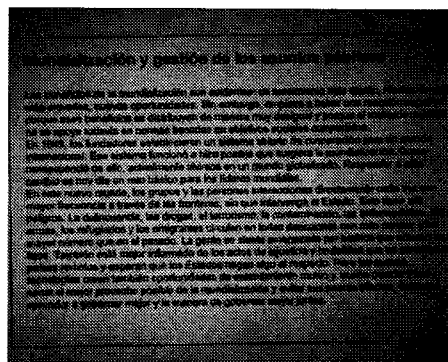
Overall goal of our system is to provide useful translation and triage results so that Soldier in the field can easily understand what is written on any information sign or on a piece of document that are written with a foreign language.

We have built Spanish to English and Arabic to English machine translation system for PDA. Translation engines runs fast enough to provide acceptable text translation less than 5 second (on the average) for text with

300 words. We have conducted offline training by using parallel sentences for documents and road signs in both Spanish and Arabic. In addition we have built transliteration tables in case the recognized word is not in our dictionaries. In addition, we have built necessary tools to support any mission specific translation and triage processing and fast new mission deployments. Currently we are conducting accuracy of our translation engines by comparing human translator and commercial translation engines. An example of Arabic translation obtain by our system, SYSTRAN and human operator are shown in Figure 10.

6 Conclusion

We have described our mobile document recognition and translation system (Kenai-II) which works on a COTS PDA with attached digital camera. Core elements of our technology are the ability to extract text information accurately and recognize each foreign text quickly with a limited resource mobile device. We have developed very fast and accurate image processing algorithm to be able to extract text information reliably from pictures and documents and recognize each character or word in the text. It does not require any special hardware and works on a COTS PDA. Once foreign text detected and characters and words are extracted, they are



System	11.4s		
Detection	1.6s		
Recognition	8.4	1164 char	12 point
Post OCR	0.5	179 word	18 pixel
Translation	0.4s		1436
Translation Table	3.1MB		Spanish
Load Translation	0.8ms		

Original Text

Mundialización y gestión de los asuntos públicos
 Los beneficios de la mundialización son evidentes: un crecimiento más rápido, niveles de vida más elevados, nuevas oportunidades. Sin embargo, empieza a haber una reacción negativa porque esos beneficios se distribuyen de manera muy desigual y porque el mercado mundial no se apoya

Translation:

GLOBALISATION MANAGEMENT THE AFFAIRS PUBLIC
 THE BENEFITS GLOBALISATION ARE GROWTH THE
 SUPPLY MORE) QUICKLY) LEVELS SHIFT THE SUPPLY
 MORE) HIGH NEW OPTION BECAUSE) CITIES STAGES
 GASES SERIOUS) HOWEVER HOWEVER BEGINS HAVE A
 REACTION NEGATIVE BECAUSE THESE BENEFITS
 REPRESENTED WAY VERY UNEQUAL BECAUSE MARKET
 WORLD SUPPORTS STILL RULES BASED OBJECTIVES
 CHARTER THIS LACK WAY) SHARED THE FOUNDING
 ESTABLISHED SYSTEM OPEN COOPERATION TO WORLD
 INTERNATIONAL

Recognition:

MUNDIALIZACIÓN GESTIÓN LOS ASUNTOS PÚBLICOS
 LOS BENEFICIOS CONTRA (MANERA MUNDIAL MUNDO
 NUNCA) NACIÓN SON EVIDENTES CRECIMIENTO LAS
 (TASA) RÁPIDA (RÁPIDO) NIVELES VIDA ARMAS
 (ESAS LAS UNAS) ELEVADOS NUEVAS
 OPORTUNIDADES SIN EMBARGO EMPIEZA HABER
 UNA REACCIÓN NEGATIVA PORQUE ESOS BENEFICIOS
 DISTRIBUYEN MANERA MUY DESIGUAL PORQUE
 MERCADO MUNDIAL APOYA CON (LOS NOS)

Figure 9: Example of results obtained with Polar Rain's Kenai-II Mobile Spanish to English Translation system. The execution times are in second and obtained with a PDA which has 64MB main memory and 624Mhz xScale processor

SOURCE:
 في اليوم الخميس (إن قهيبين)
 الإسرائيلي هدم نيل الأربعاء الخميس عشرة منازل على الأقل ختل
 عملية توغل في مخيم رفح للاجئين الفلسطينيين جنوب قطاع غزة .

Kenai-P:
 gaza [2-1] (f-15 was)the [AfAd] source ismailia official
 al-yourn thursday the army israeli destruction
 overcome Wednesday Thursday ten houses on
 [AlAQ] during process forgotten in refugee rafah
 refugees south gaza strip

SYSTRAN
 Gaza 2-1 ([aaf] in) - source securities of drawings
 reported today Israeli Thursday that the army of
 [hdm] of night Wednesday Thursday ten houses on
 little during operation penetration in camp
 Palestinian Rafah for refugees of south Gaza Strip.

HUMAN
 Gaza 02/01 (AFP) An official security source reported
 today that on Wednesday night the Israeli army
 destroyed at least ten houses during an incursion
 into Rafah Palestinian refugee camp in the Gaza
 Strip.

References

- [1] A. Jain and B. Yu Automatic Text Location in Images and Video Frames in Proc. IEEE Pattern Recognition, Vol 31, 1998
- [2] H. Li, and D. Doermann, A Video Text Detection System Based on Automated Training, ICPR, pages 223-226, 2000
- [3] E. Haritaoglu, I. Haritaoglu Real Time Scene Sign/Text Detection System For Location Awareness And Navigation International Conference on Image Processing, 2003.
- [4] I. Haritaoglu Link from Real World to Digital Information Space International conference on Ambiguous Computing, October, 2001.
- [5] I. Haritaoglu, Scene Text Extraction and Translation for Handheld Devices International conference on Computer Vision and Pattern Recognition (CVPR) December 2001.
- [6] J. Ohya, A. Shio, and S. Akamatsu, Recognizing Characters in Scene Images IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol 16 Feb, 1994
- [7] J.C. Shim, C. Dorai, R. Bolle, Automatic Text Extraction from Video for Content-Based Annotation and Retrieval In. Proc. Of ICPR, pp. 1998.
- [8] J. Spohrer, Information In Places, IBM System Journal, Vol 38, No. 4 - Pervasive computing
- [9] Anne Eisenberg, Point-Shoot-Translate into English: An Article on Newyork Times (March, 14, 2002)
- [10] Michael J. Miller, Don't leave home without your handheld translator: An Article on PC Magazine (September, 9, 2002)

Figure 10: A comparison of Arabic to English translation with our system and Systran and human translator

recognized using Polar Rain's mobile OCR engine. Polar Rain's Mobile OCR engine is capable of recognizing machine printed Latin, Arabic and Cyrillic documents and road/information signs. It is capable of generating acceptable and usable results. We have achieved more than 98% accuracy for both scripted Arabic and Latin OCR engines. Once the text is recognized, it is translated to English. Polar Rain has been developing custom domain-specific and generic mobile translation engines which is capable of translating Spanish, Arabic, Serbian to English.

Processing of Arabic Documents

A System for Discriminating Handwriting from Machine Print on Noisy Arabic Documents

John C. Femiani¹ Mariano Phielipp² Anshuman Razdan³

Abstract

A system for identifying handwritten and machine printed text from grayscale images of real-world Arabic documents is presented. The focus is on Arabic language documents because the problem is more difficult since Arabic typeset is similar to Arabic handwriting. Arabic typeset letters are connected and have similar shapes to handwritten words. Once the two types of text have been segmented it is anticipated that each can be recognized with higher accuracy using specialized treatment. We achieve positive results in segmentation by augmenting a set of attributes geared towards identification of machine printed text on binary documents (2D attributes) with attributes motivated by a 3D stroke-based interpretation of the grayscale information on documents.

1 Introduction

Our goal in this paper is to describe attributes used to distinguish typeset or machine-printed Arabic text from handwritten Arabic text in low-quality document images acquired at a resolution of 150 dots-per-inch (dpi). The authors hope that this research will ultimately be the first part of a system that provides the capability to collect, organize, and search for text in data from Arabic documents, allowing fewer language experts to accomplish more. The need to discriminate between typeset and handwritten Arabic comes from the nature of handwritten annotations: the fact that handwriting is present at all in a document can be a clue to its nature, as handwriting often contains corrections to the document or indicates the importance of the document. In spite of the fact that some tools for recognizing typeset Arabic can sometimes recognize handwriting as well, the accuracy is much worse. Therefore, special treatment may be given to the handwritten portion of a document by the use of more powerful software tools or human transcription.

The particular task motivating this research was the processing of a large dataset of documents collected en-masse and containing sensitive information. We conceived of a system whereby presence of handwriting would affect the handling of a document: All documents would be run through an Optical Character Recognition (OCR) tool to be catalogued in a keyword-searchable database. The documents could then be inspected and corrected manually based on the percentage of the document that contained handwriting (in descending order).

2 Data

The project was motivated by the influx of Arabic language documents following 9/11. Millions of documents had been acquired in digital form and the intent was to extract meaningful intelligence from the scanned documents. We received data in the form of 430 almost-randomly chosen degraded 150 ppi 8-bit gray-scale Arabic language document images that had been extracted from compressed PDF files and thus suffered compression artifacts. A significant portion of the document images was occupied by a dark region caused by the scanner bed being left open as pages were scanned. The document's page on average occupied around a quarter to a third of the entire document image, but some images contained several pages that had been placed on the scanner bed at once. The document's themselves exhibited a variety of font sizes, including images that contained only one large word as well as images with text that was only a few pixels high and thus undecipherable. Due to the classified nature of the document images, we are prohibited from publishing documents from our testing and training dataset. However, the documents we use are almost identical in nature to those available at the Iraq Research and Documentation Project (IRDP) [1] sponsored by Harvard University's Center for Middle

¹ Graduate Research Assistant, PRISM, Arizona State University, John.Femiani@asu.edu.

² Graduate Research Assistant, PRISM, Arizona State University, Mariano.Phielipp@asu.edu.

³ Research Professor and Director, Decision Theatre, Arizona State University, razdan@asu.edu

Eastern Studies, with the exception that the IRDP documents have been binarized whereas we work with grayscale images. These documents were seized from Iraq and they are not classified. The US Senate has released digital copies of over 2.4 million documents to Harvard, who has made 42,000 publicly accessible.

A key issue in processing these documents is the presence of artifacts on the page that are not printed text, images, logos, or handwriting. This includes roughly uniform perturbations of the images intensity due to inaccuracies in the scanner as well as loss in image fidelity due to an unknown lossy compression method (probably JPEG) that was applied to all of the documents in our training and testing dataset. The documents had little or no skew caused by projection since they were all scanned on a flatbed scanner, but many of them had streaks or large tonal variation in certain regions presumably due to issues with the scanner or prior damage done to the documents themselves, making accurate segmentation of text a difficult task. Speckles on the document are an issue since Arabic text is heavily accented compared to English so the dots may be important in deciphering the text, but by their nature there are so many dots in an image that they would dominate the feature-collection phase of our classifier. We resolve that issue by flagging small components and we compute a reduced feature set on only those dots which are nearby larger components. Other artifacts included tears through the pages, stains, stamps placed over the text, patterned background images, and multiple pages at arbitrary orientations in the images. When we mention noise in this paper we are referring collectively to the set of issues mentioned above.

3 Related Work

Much work has been done on separating typeset characters or words on a page as this is a first-step for OCR [2, 3]. Accurate results have been reported for English typeset documents using 2D (binary) methods, even in the presence of a significant amount of noise [4]. In their work, document elements are split into three categories: Handwriting, Print, or Noise. Three trained Fisher linear classifiers are used and their results are combined based on their classification confidences. The classification accuracy is improved by combining the Fisher classifiers with a Markov Random Field (MRF) over the word-blocks where each word is connected to the nearest word to its left and right (for print), or the four nearest neighbors in any direction (for noise). The probability of a word being print versus noise based on context is systematically estimated based on training samples, and the most likely solution is found by the discrete local minimization method of Highest Confidence First (HCF). The authors demonstrate positive results on English and Chinese

language documents, however positive results for English language documents do not always translate to good results in Arabic since Arabic typeset text and handwriting are both cursive, letters are often (but not always) connected at the baseline, and diacritical marks are difficult to distinguish from noise but are important to the meaning of Arabic writing [5]. In future work we plan to do a direct comparison of our method with the method of [4] on Arabic documents.

Stroke based models have been used to represent handwritten text with some success since they reflect the method used to produce the image [6]. A distinction is made between a stroke, trace, and a crossing such that a stroke is the concatenation of several traces to form the complete path of a pen from when it was placed on a page (called a “pen down” event) until it was removed from the page (a “pen-up” event). A trace is a portion of a stroke that does not cross itself or any other strokes, and a crossing is a region where traces meet, such as at the center of the letter “X”. Traces and crossings are identified either by skeletonizing a component or by analyzing its contours [6]. The formation of strokes from the traces allows for recovery of the drawing order of handwriting, but strokes are difficult to recover efficiently or robustly except for certain simple configurations [6-11].

The additional information in a grayscale image prior to binarization was modeled as a 3D surface in [12]. The surface was segmented into regions based on its gradient and Hessian, and the regions were used to segment strokes even in the case where one stroke would usually be lost due to blending with another. Cues from the intensity-surface of high resolution images were used for detecting stroke endpoints and stroke order in [7] which analyzes the evidence left by a ballpoint pen using forensic-style techniques to recover temporal order of strokes.

4 Our Method

The primitive element we classify is the four-way connected component. We are not concerned with breaking a component into individual letters or identifying entire words which may encompass multiple components since our goal is not to actually provide OCR. We model components using a stroke-based model that is similar to the models of [6] [10] [13] which we call a *stroke-graph* as illustrated in Figure 1. This model is an appropriate model for handwriting on a page, but one might point out that printed text was not produced by a pen-like instrument. Of course, we do not know yet which components on the page are printed or handwritten, and so we gather attributes from both a stroke-model and a raster-based model of the text. The nature of Arabic typeset is such that it appears to mimic the stroke-like nature of handwriting, unlike Latin characters such as E, K, H, F, etc.

We deal directly with the grayscale intensities as well

as with a binary mask indicating foreground versus background pixels. In this paper, however, we shall sometimes refer to “darkness” rather than intensity. When darkness is mentioned we refer to the complement of intensity. We treat the complement of the intensity of each pixel as its depth (z) measured orthogonal to the image plane, and thus we associate a set of 3D points (x, y, z) with traces in our stroke-graph.

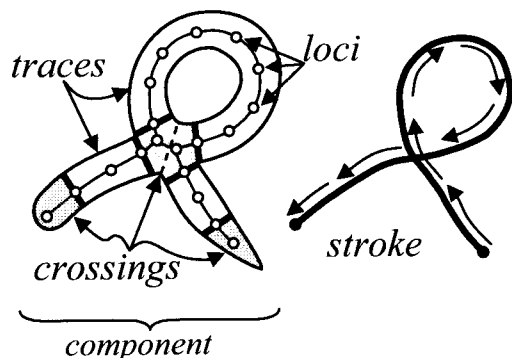


Figure 1. A single component of a stroke-graph is shown (left) with the *loci*, *crossings*, and *traces* indicated by solid arrows. A possible stroke interpretation is shown (right) with the drawing order indicated by solid arrows.

In our model, traces are represented by a chain of points we call *loci* roughly along their midline. Associated with each locus is the set of points from the image foreground that are closest to that locus. The traces are initially obtained by thinning iteratively the component [14] and then in order to compensate for noise and compression artifacts some loci are removed and others are translated according to the method explained in the paragraph accompanying Figure 1. In order to reduce the running time, we limit the number of thinning passes to the expected width of the text. Items which remain are not linear and can be removed from further consideration. Once traces are identified, the tangent to the trace at each locus is estimated by normalizing the mean of up to 5 secants centered at the tangent. Since the secants tend to get larger in magnitude as the points grow further apart, this method gives more weight to measurements taken further way from the locus, compensating for the fact that the actual positions of loci are corrupted by discretization error and error resulting from compressing the document. These tangents appear correct on visual inspection as shown in Figure 2.

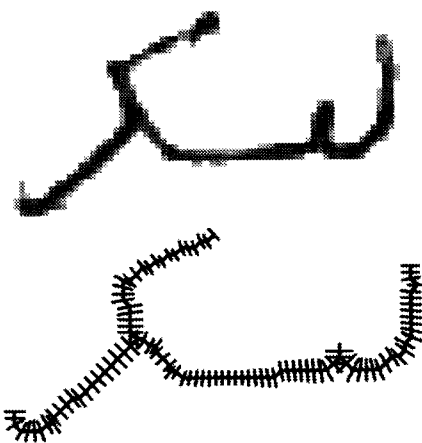


Figure 2 Top: A component with the background pixels clamped to white. Bottom: The medial axis is shown with cross-sections at each locus orthogonal to its approximated tangent.

The initial positions of the loci are not considered ideal since the midline of the trace may not be representative of the pen-path. The darkest point can not be used either since compression, noise, and the physics of writing may cause multiple darkness maxima. Our solution is to use the maximal point in a Gaussian approximation to the darkness cross-section. Each pixel associated with the locus is projected along the tangent direction. The resulting 1D function is modeled as a Gaussian, and the maximal point on the Gaussian is identified. Each locus is shifted orthogonal to its tangent so that the locus lies at the maximum of the Gaussian. The value of the Gaussian at its maximum is used as the locus’s z -coordinate.

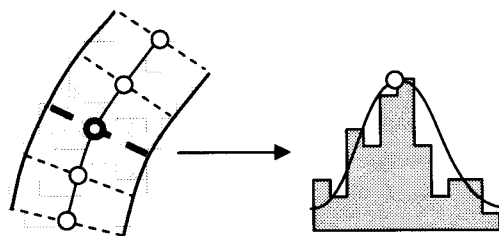


Figure 3 The pixels associated a locus (left, thick line) are project in the direction of the tangent so that their intensities form a 1D array of intensities (right, gray) that can be approximated by a Gaussian (right, solid line).

The region where multiple traces meet in a component is called a crossing, and it is initially represented by a connected set of points on the components medial axis with a valance other than two. Portions of the medial axis near the initial crossing region are removed from the neighboring traces and added to the crossing.

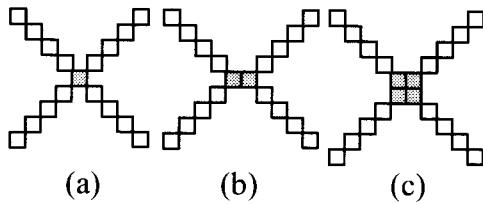


Figure 4 The ideal situation would be as in (a) where one pixel remains at the crossing (gray). The medial axis may contain clusters of points where traces meet as shown in (b) and (c).

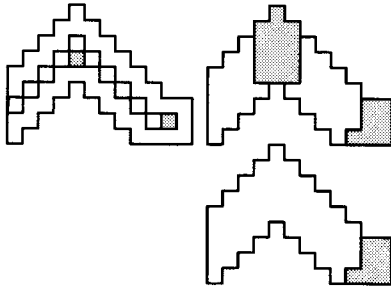


Figure 5. A crossing region (gray) is removed since it has a valance of two. The endpoint is also marked as a crossing since it has a valance of one rather than two.

Once crossings have been identified many short-traces will have entirely disappeared. If this is the case then the crossings at either end are either merged into a single larger crossing. If a merged crossing would have had a valance of two it is removed entirely and the connecting traces are joined as illustrated in Figure 5. The resulting network of traces and crossing is termed a *stroke-graph* and it is the basic unit from which many of our 3D Stroke-based attributes will be derived.

Our main concern has been determining a set of attributes based on these 3D stroke-graphs that distinguish typeset from handwritten components. We are looking for real-valued numbers that capture the difference between handwritten and printed text only. These attributes come from the binary raster information, the grayscale raster information, and from our 3D stroke based model.

4.1 2D Attributes

We began by using some attributes that were collected directly from the binary classification of pixels in the raster image into foreground and background regions. These attributes were chosen based on their reported effectiveness [4, 15] and their ease of implementation. Our goal was to allow our method to build off of methods that appear successful on binarized images. This way our tool would be able to cope with binary images or images so noisy that a

stroke-model becomes meaningless.

4.2 3D-Stroke Attributes

Attributes are accumulated for an entire component based on the mean, variance, skewness, kurtosis, Gini coefficients [16], and the mean squared sample differences of attributes collected at the trace-level. Trace attributes are in turn based on attributes measured at each locus. We begin here by listing the locus-level attributes, and then proceed up to trace and component level attributes.

The 2D position of a locus refers to its x and y coordinates on the image-plane. The mean of these positions is not kept as we do not want the components position on the page to influence the classification: however the higher moments of position are used. The stroke model we use allows for the computation of a z -coordinate at each point along the trace's path. We illustrate our method for choosing z in Figure 3. All of the previously mentioned statistics of the set of z -coordinates are used, including the mean. The radius of a locus is the minimal distance from the locus to either contour – roughly half the trace's width. Two radii are kept for each locus: they are the distances from the nearest contour to the left and right of the locus as we traverse the contour⁴. The full set of statistics is used on the set radii. The error measures the discrepancy between the Gaussian approximation and the actual cross section and measured as the mean squared error taken from Figure 3. We only accumulate the mean of this quantity. The tangent angle of a locus is measured up from the detected horizontal axis of the page. Tangent angles are not accumulated: rather their respective tangent vectors are accumulated and converted to angles afterwards. The mean tangent is excluded from the set of attributes used to classify a component, but the mean difference between angles is used. Finally the number of loci in a trace, and the number of traces in a component are used as attributes as well.

One final source of attributes was added as a simple way of incorporating some of the components context within the document. The statistics kept for each component are also accumulated over a 1.5-inch neighborhood of each component and included as additional features. We achieved our best result by training two classifiers using the same algorithm (jRip) [15], but we derived additional features for a second classifier by using the results of the first classifier to determine if a 1.5 inch neighborhood is mostly hand or mostly print. We refer to the classifier that is run first

⁴ The left and right are ill-defined since we do not know the direction in which the contour was traversed. Our implementation currently always begins in the leftmost column of the topmost row in which the component occurs.

as classifier 1, and the classifier based on its result as classifier 2.

4.3 Classification

We use the jRip propositional rule learner [17] based on the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) algorithm first proposed by Cohen [18] in order to generate a classifier based on our attribute set and training data. We chose to use jRip over other methods because it generates a set of rules that are often simple enough for a human to understand. Additionally, it produced classified as many elements correctly as some of the most popular methods such support vector machines and neural networks (using WEKA). Another motivation behind using this classifier is that while it takes some time to train, it can classify components very quickly. Rather than explicitly model context as a MRF in the document, we collect all neighbors whose bounding boxes are within a predefined distance of each-other of 1 inch or 150 pixels. We trained two classifiers, one on features computed at each component, and the other with a feature set augmented by fraction of each component's neighbors classified as hand or print by the first classifier.

In order to generate training and testing data for our experiment, we generated an interactive tool for manual entry of ground-truth for components in the documents. We recorded each images filename and the coordinates of the darkest pixel of each component in the image along with information on the component as to whether it appeared to be handwriting, printed, image, noise (of various sorts), an embedded image, border, stamp, or a merged component consisting of both printed and handwritten parts. The tool allowed the user to select from a list of 15 categories and label components by either clicking on them, or by drawing a region around them. In this manner 47,287 components were selected of which were 20,491 printed, 20,567 were handwritten or contained components that were hand and print merged together due to overlap⁵, and the remaining were items we did not wish to include in either category (such as holes, noise, borders, or logos).

Our method was implemented in C++ and compiled with all speed-optimization flags set. It took an average 4.3 seconds per document on a Pentium Xeon 3.2 GHz computer with 2 Gb of RAM. Of these, on average 2.72 seconds were spent on forming the initial image segmentation, growing connected components, finding

⁵ For the purposes of this project, merged hand & print are lumped together with handwritten components since are difficult to OCR and thus require the same extra attention as handwritten components. Also overlap tends occur between handwritten words rather than hand-over-print in our test data.

skeletons and creating the stroke-graph. The remaining 1.31 seconds were spent collecting features and evaluating the 2 sets of decision rules generated by jRip. Our results are presented in **Table 1** using a confusion matrix. Each column corresponds to the training label of a component, each row corresponds to the class predicted by our classifier, and each entry in the matrix shows the number of instances trained according to the column and predicted according to the row. We were able to identify 94.56% of the components correctly⁶, and we identified 94.31% of all handwritten items in our test set and 94.8% of all printed items.

Table 1 Results of our classification of the entire feature set, presented as a Confusion Matrix.

Predicted Typeset	Predicted Handwritten	
19325	1166	True Typeset
1069	19498	True Handwritten

5 Conclusions and Future Work

We have presented our system for discriminating between Arabic handwriting and typeset text. We have achieve positive results on our training and testing corpus, and conclude that incorporating 3D features based on a stroke-model of the text derived directly from grayscale images can improve our ability to separate printed from handwritten text. In future work we would like to compare our method carefully with the method of [4] and explore other ways of dealing with document layout and noise in the types of documents we see. Since handwritten or mixed components appear to have been successfully identified, we would like to enable queries for words through handwritten portions of text using a shape-based word-spotting algorithm. Also we would like to extend out ability to capture strokes and resolve crossings. Our current stroke based model solves simple examples of crossovers, but suffers from some extreme cases of ambiguity cause by sub-pixels attributes in text or image degradation due to compression.

Acknowledgements

The authors would like to thank the Partnership for Research in Spatial modeling (PRISM) and other sponsors, as well as the reviewers for their helpful comments.

⁶ The tool "WEKA" could not process all 40,000 entries and so it was trained with roughly 50% of the samples and reported accuracy greater than 96% on that subset.

References:

- [1] M. Kanan, R. Rend, H. Mneimneh and R. Rabil "The Iraq Research and Documentation Project (IRDP)," <http://www.fas.harvard.edu/~irdp/>, September 15, 2005.
- [2] Zheng, Liu and Ding "Single-character type identification," *Document Recognition and Retrieval IX* vol. 4670, no. 1, pp. 49-56, 2001.
- [3] Pal and Chaudhuri "Automatic separation of machine-printed and hand-written text lines," *ICDAR '99* pp. 645-648, 1999.
- [4] Y. Zheng, H. Li and D. Doermann "Machine Printed Text and Handwriting Identification in Noisy Document Images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* vol. 26, no. 3, pp. 337-353, 2004.
- [5] A. Amin "Off-Line Arabic Character Recognition: A Survey," *ICDAR97* pp. 1997.
- [6] R. Plamondon and C. M. Privitera "The Segmentation of Cursive Handwriting: An Approach Based on Off-Line Recovery of the Motor-Temporal Information," *IEEE Transactions on Image Processing* vol. 8, no. 1, pp. 80-91, 1999.
- [7] Doermann and Rosenfeld "Recovery of temporal information from static images of handwriting," *CVPR '92* pp. 162-168, 1992.
- [8] I.S. Abuhaiba and P. Ahmed "Restoration of Temporal Information in Off-Line Arabic Handwriting," *PR* vol. 26, no. 7, pp. 1009 1993.
- [9] Bunke, Ammann, Kaufmann, Ha, Schenkel, Seiler and Eggimann "Recovery of temporal information of cursively handwritten words for on-line recognition," *ICDAR'97* vol. 2, pp. 931-935, 1997.
- [10] Hew and Alder "Strokes from pen-opposed extended edges," *ICONIP '99* vol. 3, pp. 1155-1160, 1999.
- [11] Kato and Yasuhara "Recovery of drawing order from single-stroke handwriting images," *PAMI* vol. 22, no. 9, pp. 938-949, 2000.
- [12] L. Wang and T. Pavlidis "Direct Gray-Scale Extraction of Features for Character Recognition," *PAMI* vol. 15, no. 10, pp. 1053-1067, 1993.
- [13] Nishida "An approach to integration of off-line and on-line recognition of handwriting," *Pattern Recognition Letters* vol. 16, no. 11, pp. 1213-1219, 1995.
- [14] Lam, Lee and Suen "Thinning methodologies-a comprehensive survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* vol. 14, no. 9, pp. 869-885, 1992.
- [15] Y. Zheng, H. Li and Doermann "Text identification in noisy document images using Markov random model," *ICDAR'03* pp. 599-603, 2003.
- [16] G.J. Glasser "Variance Formulas for the Mean Difference and Coefficient of Concentration," *Journal of the American Statistical Association* vol. 57, no. 299, pp. 648-654, 1962.
- [17] I.H. Witten and E. Frank "Data Mining: Practical machine learning tools with Java implementations," 2000.
- [18] W.W. Cohen "Fast Effective Rule Induction," *Proc. of the 12th International Conference on Machine Learning* pp. 115-123, 1995.

NLP-Enhanced Exploitation of Degraded Arabic Texts

Michael Shalev

Encyclopædia Britannica,
Natural Language Technologies Division
311 S. Michigan Avenue,
Chicago, IL 60604
michaels@eb.com

Abstract

Arabic poses significant challenges to automatic exploitation due its highly inflectional nature (stems can have hundreds or even thousands of inflected forms), and its consonantal orthography (most vowels are omitted), which causes high ambiguity rates. These challenges are even more pronounced when dealing with texts that are degraded.

Arabic OCR character recognition accuracy is adversely affected by Arabic's cursive script, usage of diacritic dots, and the relatively large number of letter forms (each letter may have several allographs depending on the context of adjacent letters).

We applied a context-sensitive morphological analysis system to degraded Arabic texts received in lattice format to examine the potential contribution of linguistic analysis to text exploitation of degraded Arabic documents. The enhancement was measured at various levels of character recognition accuracy (corresponding to accuracy levels typical of OCR output). The resulting levels of word recognition and entity detection were compared to the baseline levels of the generated degraded texts.

The results show improvement in the number of recognized words and entities in the linguistically processed texts compared to the raw degraded documents at all levels of input letter accuracy.

1 Introduction

1.1 Arabic Morphology

Arabic¹ morphology is commonly viewed as being based on underlying consonantal roots (*jd̄r*, جذر) that undergo vowel changes, and in some cases consonantal

¹ Arabic here refers to Modern Standard Arabic (MSA).

insertions and deletions, to create inflected and derived forms [5]. For example, the root /ktb/كتب combined with the vocalic pattern CaCaCa² (where 'C' represents a consonant and 'a' represents a vowel) derives the verb *kataba*³ كتب 'to write'. This derivation may be further inflected into forms indicating semantic features, such as number, gender, tense etc.: *katab-tu* كتبت 'I wrote', *katab-ta* كتبت 'you (sing. masc.) wrote', *katab-ti* كتبت 'you (sing. fem.) wrote', *?a-ktubu* اكتب 'I write/will write' etc.

In addition, the definite article, as well as several prepositions and conjunctions are considered prefixes and not autonomous words; additionally, pronominal objects and possessive pronouns are suffixed to verbs and nouns, respectively.

The structure of Arabic morphology results in each word having a very high number of inflections, which may radically change the basic form of the words. Accordingly, the total number of Arabic inflected forms is estimated to be 70 million [1], [2].

1.2 Arabic Orthography

Arabic orthography is primarily consonantal - most vowels are not written. There is a supplementary system of vowel and gemination diacritics (written above, under and beside the text), - but these are omitted in most written texts⁴. As a result, the rate of form ambiguity, i.e. ratio of morphological analyses per

² This pattern is known in Arabic grammar as وزن فَعَلَ.

³ Latin transcription of Arabic examples and terms follows English spelling conventions. The character: "?" denotes a glottal stop; and "ð" denotes the sound "th" in "this".

⁴ This supplementary marking system is used in this document for some Arabic examples, to emphasize the difference between Arabic words.

given string, is considerably higher than in most languages.

2 Arabic OCR Challenges

In addition to morphological and orthographical features that complicate Arabic text exploitation in general, (e.g. search, translation and entity extraction), Arabic script and letter shapes (printed as well as handwritten) present additional challenges that are specific to OCR, including:

- Cursive Script: most characters are connected and their shape is context (position) sensitive: a letter may have different shapes when appearing in isolation, in word-initial, -median, or -final position; letters may have as many as 4 shapes.
- Word elongations (tatwil تطويل) and ligatures (letter combinations, such as lam-alif “لا”) expand the number of possibilities.
- Diacritic marks serve to optionally mark vowels as well as to distinguish between several of the Arabic letters, where they are obligatory. Dust, dirt, and specks may be confused with diacritic marks, further complicating letter recognition.
- Punctuation marks are often omitted and cursive script is sometimes broken within a word, making the distinction between word boundaries and inner word spaces more difficult, and complicating segmentation.

Typically, OCR software generates several character candidates per image segment, each designated by a confidence score; such OCR internal structures are referred to as lattices.

While statistical and language-independent OCR strategies may suffice to provide acceptable accuracy levels for some languages (e.g. English) in which the above considerations are absent, or present to a lesser degree, when dealing with Arabic texts, whether printed, or even more significantly, handwritten, the accuracy levels offered by such methods decrease, and often may not yield sufficient accuracy to be of practical value.

The purpose of this study is to examine the contribution of applying linguistic post-processing to enhance the exploitation of Arabic OCR output. Exploitation enhancement is achieved by identifying correct and appropriate words within OCR output lattices, thereby making the texts more searchable, and therefore more suitable for automatic categorization and routing processes. Enabling such a linguistic-based accuracy enhancement requires a robust NLP system that can identify and score all 70 million inflected forms that appear in Arabic, and be able to propose the

most appropriate word represented by the OCR word lattice.

While fully accurate OCR of Arabic handwriting will remain difficult to achieve, it should be noted that:

- Partial handwriting recognition, even at low accuracy rates, is potentially of great value.
- Partial recognition may allow identifying important keywords in documents.
- Improvement in word recognition over character lattices at various accuracy levels can contribute significantly to automatic processes.

3 Natural Language Processing

In pursuit to enhance and exploit degraded Arabic documents, several approaches were examined, mainly character and word n-grams [4], [6], light stemming (algorithmic prefix and suffix removal) [3] and closed-end partial lexicons.

The current study presents an approach based on applying a robust stemming/morphological analysis system to exploit and enhance degraded Arabic documents.

3.1 Morphological Analysis

The Britannica Morphological Analyzer (BMA) for Arabic was used in this study. BMA identifies all Arabic word forms, breaking down inflected forms into their base form (stem) and inflectional morphemes. BMA applies scoring based on frequency measures of each stem, as defined in the BMA lexicon, a frequency measure of the morphological variant, and a measure calculated by context-sensitive analysis, which checks for validity of syntactic phrase structure.

When BMA is applied as a stemmer in search engine indexing systems, the score is used to ensure that the correct stem is indexed in cases of ambiguity, so that only inflections relevant to the query (i.e. the stem appearing in the query) are returned, and not unrelated forms that happen to share the same spelling with one of the inflected forms of the query terms.

When applied to degraded text, BMA's scores are used differently: rather than using the scores to choose between *alternative morphological analyses* of a given string, the scoring system is used to choose between different *strings*, representing different word forms proposed for a given segment of the degraded text.

3.2 Named Entity Recognition (NER)

Named Entities (NEs) refer to proper names (e.g. people, places, organizations, expressions of time, etc.), and hold a special significance in text exploitation due to their high content value. NER attempts to identify and classify Named Entities by referring to NE

lexicons. Cross-language NER translates entities identified in Arabic into English.

Performing NER by linguistically enhancing degraded input intends to provide more valuable results from the perspective of text exploitation. EntX, Britannica's cross-language NER system, was used in this study.

4 Data

40 typical documents from Arabic media web sites (Al-Hayyat, Elaph, Alquds, etc.) were selected. The length of these documents ranges from 150 to 200 words, yielding a sample corpus of over 7,000 words.

The documents then underwent automatic degradation to four levels of character accuracy typical of handwritten OCR output (see section 5.1 below), yielding 160 sets of output lattices (four for each original document), and a total of over 28,000 words.

5 Method

5.1 Automatic Degradation

To simulate OCR results at different levels of accuracy, an automatic text degradation program was developed, modeled on OCR lattices of a sample corpus of handwritten documents. The program takes as input an Arabic text document and the desired level of degradation (defined as the letter accuracy⁵ level), and returns sets of word lattices for the identified words/strings in each degraded document.

The degradation process has several parameters:

- **Character accuracy:** the probability of the character being correctly identified by the OCR process (i.e. being shown as the topmost character candidate per position)
- **Lattice depth:** the number of character candidates for a given character (0-5)
- **Lattice quality:** the probability of the correct character appearing in the generated lattice, providing it wasn't selected as the first candidate
- **Segmentation:** lattices are either split, or separate lattice joined to model incorrect identification of word boundaries. The segmentation quality reflects the percentage of words correctly segmented)

The lattices are populated with random characters and random relational confidence scores (aside from the topmost character as defined by letter accuracy, having the top score, 0).

Note that the fact that lattices are populated randomly makes the output somewhat more difficult to exploit, as compared to OCR output, where scores of correct ground truth character candidates in the lattice are likely to be better (lower) than those of incorrect characters.

Each document is degraded to four levels of character accuracy: 30%, 45%, 60%, and 75%, yielding 4 documents for each original text, in addition to the original, which serves as the ground truth for the degraded documents.

Given the character accuracy, the lattice quality and segmentation quality were adjusted to reflect typical OCR results, in a manner illustrated in table 1.

Our definition of character accuracy follows the one proposed by Rice, Kanai and Nartker [7], based on the Edit Distance algorithm (known as the Levenshtein algorithm).

The following table describes the character accuracy, lattice quality and segmentation quality of the degraded documents.

Table 1: Lattice quality and segmentation quality per Letter accuracy.

Character accuracy	Lattice quality	Segmentation quality
30%	55%	25%
45%	65%	65%
60%	75%	80%
75%	90%	90%

The output (i.e. 'identified' words) of each degraded document was compared to the ground truth to determine the word accuracy achieved in the simulated OCR output per document (for each letter accuracy level), which serves as the baseline.

5.2 NLP Processing

5.2.1 Word-Internal Lattice Enhancement

Each sequence of characters in the lattice is compared with a set of patterns reflecting typical noun and verb prefixes, suffixes, and internal patterns. In cases where a pattern is partially found, this process may introduce an additional character candidate to the lattice. Consider the following table:

⁵ The probability of the correct letter appearing in the OCR input in this position (compared to the ground truth).

Table 2: Example Lattice⁶ of البحرية ("the marine"). Correct letters appear in red. Optical scores appear under letter candidates - lower scores are better.

String	Letter	ا	ا	و	ب	م	ي	ة
Option 1	Score	0	0	0	31	0	0	0
String	Letter	ا	ئ	ب	ه	ر	أ	ة
Option 2	Score	85	20	35	58	10	51	35
String	Letter		ص	ظ	م	ث	و	ر
Option 3	Score		57	4	70	15	32	70
String	Letter		ء	و	ح	ة	س	ى
Option 4	Score		216	76	75	138	183	78
String	Letter		ل	ه		س	س	ف
Option 5	Score			180		20	200	21
Output		ا	ل	ب	ح	ر	ي	ة

In string option 1 above (the first row) there is a word-initial sequence of two alifs ("اا"). The word-internal processing introduces a candidate lam ("ل") in the second position as the sequence "ال" is the Arabic determiner, which has a very high likelihood of appearing in word-initial position. This expands the set of strings that may be generated from the lattice. Following this process, the set of all possible strings or character combinations resulting from each lattice (generally ranges from hundreds to thousands) is generated. These strings are then subject to word level linguistic processing (see following section).

5.2.2 Word Level Linguistic Processing

All generated strings first undergo morphological analysis to determine which, if any, of the strings are morphologically well-formed. The stems of the set of well-formed strings are then subjected to lexical look-up to determine if they are known words. The lexicon contains statistical data regarding distribution of stems and inflected forms. If sufficient context of neighboring words exists, context-sensitive weighting is applied to the word candidates. Each step of the word level linguistic processing contributes to the linguistic score accorded to the word (see below).

5.2.3 Linguistic Scoring

A *linguistic score* is determined per string (as opposed to the *optical score* given to each *character candidate* by the OCR process), which considers the following factors: frequency and distribution probability of lexical stems and inflection patterns; and

⁶ 45% character accuracy; purple cell contains a character introduced by linguistic post-processing

contextual weighting (if a sufficient context of neighboring words exists).

Thus the best linguistic scores are assigned to words that are recognized as belonging to widespread stems with common inflections which fit well in their syntactic context. Words that are recognized by BMA but are found to be derived from less frequent stems, and/or feature rare morphological structures or are inappropriate with their context are given higher (worse) scores. Words that are not recognized at all are discarded.

Table 3: output for three different strings considered for a certain lattice (given in phonetic transcription). Lower scores signify better confidence.

String candidate	Linguistic score	Gloss	Details
<i>Alsban</i> <i>almhtrfyn</i> (الشبان المحترفين)	12	<i>The young experts</i>	valid words, correct phrase structure - well formed agreement
<i>Alsbak</i> <i>almhtrfyn</i> (الشباك المحترفين)	23	ungrammatical (* <i>experts the window</i>)	valid words, incorrect grammatical context
<i>Alsblk</i> <i>almhtrfyn</i> (الشبيك المحترفين)	-	word discarded	invalid word

A composite score of the *linguistic score* of a word and an *optical score*⁷ for this word is calculated and the string with the highest composite score is returned. The composite score is calculated by 75%-25% weighting in favor of the linguistic score (see further discussion below regarding weighting considerations).

6 Results

The word accuracy and named entity (NE) accuracy of each document was determined by manually comparing it with the respective ground truth. The 160 degraded documents were then subject to NLP processing (see section 5), providing 160 linguistically enhanced documents. These too were manually compared to the ground truths, and word and NE

⁷ The optical score of the string is calculated as the average optical score of the characters the string consists of.

accuracy were determined. The results appear in figure 1 below.

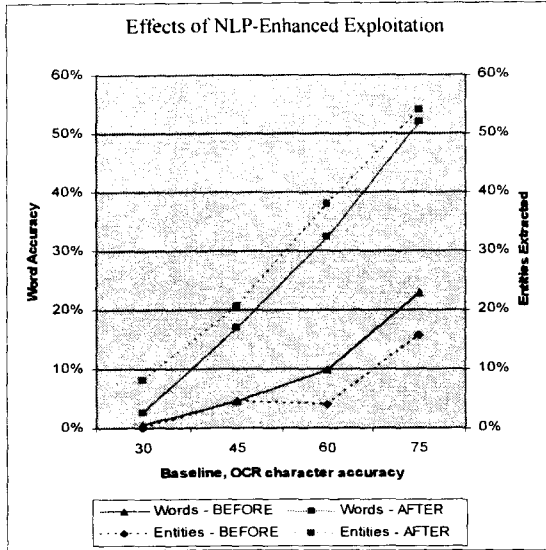


Figure 1: Effects of NLP processing of degraded documents: word and entity accuracy before and after NLP processing

The solid blue line represents the base level word accuracy level for each of the four degradation levels (30%, 45%, 60%, and 75%), while the dashed blue line represents the NE accuracy at these levels.

The solid magenta line shows NLP-enhanced word accuracy for each degradation level, and the dashed magenta line shows the NLP-enhanced NE accuracy. The following tables show the word- and NE-recognition for the four levels of character accuracy.

Table 4: Word-recognition as a function of character-accuracy

Character accuracy	Word recognition	
	Non-enhanced	NLP-enhanced
30%	0.5%	2.6%
45%	4.4%	17%
60%	9.8%	32.2%
75%	22.9%	51.9%

Table 4 above shows an improvement from 0.5% to 2.6% for highly degraded documents, to from 23% to 52% for the least degraded documents.

Table 5: Named Entity-recognition as a function of character-accuracy

Character accuracy	Named entity recognition	
	Non-enhanced	NLP-enhanced
30%	0%	8.2%
45%	4.2%	20.6%
60%	4%	37.9%
75%	15.7%	54%

Table 5 above shows an improvement from 0% to 8.2% for highly degraded documents, to from 15.7% to 54% for the least degraded documents.

7 Discussion

The results show significant improvement both in word and in named entity (NE) recognition accuracy following linguistic post-processing at all character accuracy levels.

While word accuracy is higher than NE accuracy in non-enhanced results, NE accuracy is higher than word accuracy in NLP-enhanced OCR. Average word length in Arabic is 4.63 characters, while average NE length is 6.4 characters⁸. This characteristic may help explain the different patterns observed between NLP-enhanced and non-enhanced entity accuracy versus word accuracy: since entities are by and large longer than general words, the probability of an incorrectly identified letter is greater for entities than for words; however, when applying NLP-enhancement, string length contributes positively to the probability of successfully identifying the word, as there are more cues in the input.

From a practical perspective of trying to increase the value of OCR to text exploitation, increasing recognition levels of entities, which are of high content value, is more valuable than the 'across-the-board' increase achieved in word accuracy.

NER was applied by looking at a pre-defined set of entities, a subset of the words identified by BMA. An alternative approach would be to actively search for user-defined entities, such as certain proper names, and trying to match them against given lattices; this may produce more false-positives, but may limit the documents with these specific entities that would be overlooked to a minimum.

This study used a pre-defined weighting allowing 75% to the linguistic score, and a 25% weight to the optical score. This was based on preliminary tuning, however, further work needs to be carried out to find the optimal balance between the two scores, which we

⁸ These values are based on a sample corpus of documents/articles collected from Arabic media web sites.

expect to be dependent on type of material and OCR quality.

Appendix A shows an example Arabic text, in raw degraded form and in linguistically enhanced form. Correct words and NEs are marked in both documents.

Appendix B shows an XML excerpt for a word processed by BMA, and shows the metadata that may serve other exploitation processes.

8 Acknowledgements

I would like to thank the team at Encyclopædia Britannica's Natural Language Technologies Division.

References

- [1] K. R. Beesley, Finite-State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans in 2001. In *ACL Workshop on Arabic Language Processing*. Toulouse, France, 2001.
- [2] Y. Choueka, personal communication.
- [3] K. Darwish and D.W. Oard, Term selection for searching printed Arabic. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. Finland, 2002.
- [4] S. Harding, W. Croft and C. Weir, Probabilistic Retrieval of OCR Degraded Text Using N-Grams. In *European Conference on Digital Libraries*. Pisa, Italy, 1997.
- [5] J.J. McCarthy, *Formal Problems in Semitic Phonology and Morphology*. (Garland, New York; London, 1985).
- [6] S.H. Mustafa and Q.A. Al-Radaideh, Using N-grams for Arabic Text Searching. In *Journal of the American Society for Information Science and Technology*, Volume 55, Issue 11, (Wiley Periodicals, 2004).
- [7] S.V. Rice, J. Kanai and T.A. Nartker. An Evaluation of OCR Accuracy. In *The Second Annual Symposium on Document Analysis and Information Retrieval*. Las Vegas, Nevada, 1993.

Appendix A – Sample Arabic Document

Figure 2 shows an example Arabic text degraded to 60% character accuracy.

قذك ظغف جنودم بشمب البظلية اعهمريفاًس يغ عزذنيو مالبايا قسا
الراماصو عسجي يلعرهق.وقتل اخجيش اشاماطاي فس ليحي بن خفي
جعاد غتلول لفن ظرتهلهم عربتهم تتيو ع ضاجفو خزلت عصى اكسد عاغ
بلجدينء التظتعو ثعسرا لامطبتين مس الضاش.قالءا الدجتن اث ظهريا
ءادا مظ قواء جضماضينم تتع برصاخ متبايليت بفسلدينة.ويعد الهنوي ءت
ملقبتي وضال ضسبغ صحفي يوقب خلال زلا اتسنن محل قئا وليبد
ضلاءبير من الفتقى كذن اضرلو مءاي الفجرية وجف نفس
وذننظقة عوقمل لراذ أركخيفن ان تحض الأقطلي يلعاون الى سضع
كريات طابقه من بلمتقرات سي نفعك تحادط بوي يصكنهم نوءر
للعباح الأءرة حقلغ نلظبببائ التقيتمتلان عماءه بف مشاة هءبايق
اساريزوة قف يدازي خءى كوا تلحتس عظمري طي انظنور قببش
على جابخ ناطعلق وضم نر أريترم قرك اسردام.ومنئ انترى
اخامبيكي للعباء عام لذ03 قءه خدى ذذائل في المصعلى ذىءه جهتي
امريكلسقط مابءة اقمظظي 0ء4 روف صخش نسبهه حطري كل5
طولومترا طن الراصسى بغداد.وفهش ماوت مو زلكتيز بين
غدمسوقين عن روح قننءم أي الرمادي بهم وجدنا ان نسزوي
اتمقومة بلمءاءة طاعبل ملمطجاء بمقنة الهنذجءكوقالت كءضوطيح
ضانمي معخذلة ضي بت سح شحسعر اف ان صلجكش نلاضتكنض قلدم
ككك تقاصيا قمبظ عض قلعدلات الهى وتعرض طمة سلجنود ضءةلين
نلك طفنه نحاحض لحبلمتمف

Figure 2. Non-enhanced OCR output (16 correctly identified words appear in brown)

Figure 3 below shows the result of linguistic post-processing applied to the document; an additional 47 words were recognized, of which 12 are named entities (versus none in the original degraded document).

فكك سقق جنود من عذاب التسلية اعهمريفاًس في سادسين منفصلين
فيك الرمادي عسجي يلعرهق.وقتل الجيش اشاماطاي في ليحي بن
خفي جزا قتلوا لذي ظرتهلء عربتهم بعوة ضاجفو زرع عصى
اكسف عربي بلجدينء الخط تعد مسرا للمسلحين مس الضاش.قالءا
الشجون ان ظهري سائدا من قواء جضماضيفم تبع ارضاء سجنين
بفسلدينة.ويعد الهوء او كلتاني وضال أسبوع الذي ينفع خلال زلا اتسنن
محل هذا أربع ضلاءبير من الفترى كذن اضرء مشاة القارية وجف
نفس وذننظقة عوقمل لرن أركخيفن ان تحض الأقطلي يلعاون الى
سرع كريات كئير من المتفجرات المنفجرات نفعك محدد بول يمكنهم
تفجير العربات الأءة وفرجى نلظبببائ التقيتمتلان كمصر بل مشاة
البطريق اساريزو قف يدازي ايضا او الخميس لحضاري طي انظنور
قببصر على جانب اصبعيه وضم فر عرقتهم غرب اسردام.وفنى فلنزر
اخامبيكي للعراق عام لنمس قتل على الاقل في المصار ذى جهتي
امريكلسقط عابء اقتداءي زها جوي شخص رجه حواري كلت
كيلومترا طن العاصفة بغداد.وفهش مالت من الماريزو بين غدمسوقين
عن روح النظم أي الرمادي بهم وجدوا ان مستوى المقاومة
بالمضافة يعادل الموجود بمبينة الهنذجءكوقالت كبر صدين غانمي
معاقل ضد بت ض شحسعر اق ان الجيش نلاضتكنض فلجم كفل
تقاصيل فلبب عض الهجمات الهى وتعرض لم الجنود مقلين نلك طفنه
محوالا لحبلمتمف

Figure 3. BMA-enhanced OCR Output (45 newly recognized words in blue, NEs are highlighted)

Figure 4 below shows the document returned by Britannica's EntX server, containing cross-language NER. The English entity digest appears at the head of the document, and Britannica's ETL (Embedded Translation Layer) provides a hidden layer of translation for all words/phrases that is displayed by placing the mouse (cursor) over any word/phrase (see middle of document by mouse cursor in figure 4).

(c) Encyclopaedia Britannica Inc 2004-2005	
Page Entity info	
Nationality/Affiliation	Arab
Weapon	explosives
Military soldier, army	
Military Event	explosion, attacks
Place	Egypt, Iraq
Time expression	Thursday

مكك سقق جنود من عذاب التسلية اعهمريفاًس في سادسين منفصلين فيك الرمادي عسجيبلعرهق.وقتل الجيش اشاماطاي في ليحي بن خفي جزا قتلوا لذي ظرتهلء عربتهم بعوة ضاجفو زرع عصى اكسف عربي بلجدينء الخط تعد مسرا للمسلحين مس الضاش.قالءا الشجون ان ظهري سائدا من قواء جضماضيفم تبع ارضاء سجنين بفسلدينة.ويعد الهوء او كلتاني وضال أسبوع الذي ينفع خلال زلا اتسنن محل هذا أربع ضلاءبير من الفترى كذن اضرء مشاة القارية وجف نفس وذننظقة عوقمل لرن أركخيفن ان تحض الأقطلي يلعاون الى سرعة كريات كئير من المتفجرات نفعك محدد بول يمكنهم تفجير العربات الأءة وفرجى نلظبببائ التقيتمتلان كمصر بل مشاة البطريق اساريزو قف يدازي ايضا او الخميس لحضاري طي انظنور قببصر على جانب اصبعيه وضم فر عرقتهم غرب اسردام.وفنى فلنزر اخامبيكي للعراق عام لنمس قتل على الاقل في المصار ذى جهتي امريكلسقط عابء اقتداءي زها جوي شخص رجه حواري كلت كيلومترا طن العاصفة بغداد.وفهش مالت من الماريزو بين غدمسوقين عن روح النظم أي الرمادي بهم وجدوا ان مستوى المقاومة بالمضافة يعادل الموجود بمبينة الهنذجءكوقالت كبر صدين غانمي معاقل ضد بت ض شحسعر اق ان الجيش نلاضتكنض فلجم كفل تقاصيل فلبب عض الهجمات الهى وتعرض لم الجنود مقلين نلك طفنه محوالا لحبلمتمف

Figure 4. EntX processing of NLP-enhanced text, showing the entities highlighted by category, with their English equivalents appearing in the entity digest header.

Appendix B – BMA XML Excerpt

The following XML document shows an excerpt of the morphological and other linguistic metadata added by the Britannica Morphological Analyzer (BMA).

```
= <root>
  = <word>
    = <input>
      - <![CDATA[وكتبه]]>
    </input>
    = <result>
      <stem>كتب</stem>
      <transcription>kataba</transcription>
      <partOfSpeech>verb</partOfSpeech>
      <root>كتب</root>
      <conjugation>1</conjugation>
      <category />
      = <rawTranslation>
        - <![CDATA[ to write]]>
      </rawTranslation>
      = <prefix>
        <string_prefix />
      </prefix>
      = <suffix>
        <string_suffix>ه</string_suffix>
        <person_suffix>3</person_suffix>
        <number_suffix>singular</number_suffix>
        <gender_suffix>masculine</gender_suffix>
      </suffix>
      <score>85</score>
      = <inflection>
        <vocalizedWord>وكتبه</vocalizedWord>
        <inflectionTranscription>wakatabahu</inflectionTranscription>
        <person>3</person>
        <number>singular</number>
        <gender>masculine</gender>
        <state />
        <tense>past</tense>
        <voice />
      </inflection>
      = <morphTranslation>
        <conjTranslation>and</conjTranslation>
        = <inflectionTranslation>
          - <![CDATA[wrote]]>
        </inflectionTranslation>
        <objectTranslation>it</objectTranslation>
      </morphTranslation>
    </result>
  </word>
</root>
```

Figure 5. BMA XML Output for the Arabic Word وكتبه (“he wrote it”)

Initial Results in Offline Arabic Handwriting Recognition Using Large-Scale Geometric Features

Ilya Zavorin

Eugene Borovikov
CACI International Inc
4831 Walden Lane
Lanham, MD 20706

Mark Turner

{izavorin, eborovikov, miturner}@caci.com

Abstract

This paper presents preliminary results that we obtained during the first “proof-of-concept” stage of our on-going research on offline Arabic handwriting recognition. Our method relies on the Arabic script’s large-scale features (e.g. loops, cross- and end-points) and employs a double-layered discrete Hidden Markov Model (HMM) for character and word recognition. This approach does not require manual segmentation of words into characters during training or testing. The prototype system uses manual feature extraction and yields high recognition rates on synthetically generated words.

1 Introduction

Although there have been extraordinary advances in automatic recognition of machine-printed Arabic text over the last several decades, the problem of automatic offline recognition of handwritten Arabic script remains largely unsolved. As Edelman et al. [3] point out, the two main reasons for the difficulty of this task are

- character segmentation ambiguity; and
- character shape variability

Arabic script is inherently cursive, meaning that a considerable part of a word image may consist of ligatures that connect consecutive characters. These ligatures may often be interpreted as parts of letters giving rise to multiple equally plausible segmentations. Letter shapes are position-dependent. Arabic text contains numerous diacritics, overlaps, and casheedas. Furthermore, even when written by the same person, the same characters may change both geometry and topology (see, for example, Figure 1 taken from the corpus described by Pechwitz et al. in [8]). All these characteristics, combined with other language-independent irregularities typical in handwritten text, make Arabic handwriting recognition problem extremely difficult.

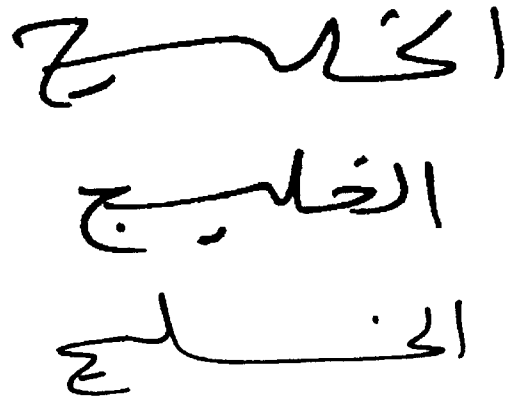


Figure 1: Same word written by the same writer

The core of a complete ICR system is its recognition unit. According to Steinherz et al. [14], one way of classifying recognition algorithms is by the size and nature of the *lexicon* involved, as well as by whether or not a *segmentation* stage is present. Lexicons may be

- small and specific containing about 100 words,
- limited, but dynamic, which may go up to 1000 words, and
- large or unlimited.

Segmentation-based methods take a sequence of primitive segments derived from a word image together with a group of possible words, and attempt to concatenate individual segments to best match candidate characters. In the segmentation-free case, no attempt is made to separate the image into pieces that relate to characters, although splitting into smaller pieces is still possible, followed by generation of a sequence of observations. In both cases, algorithms try to reproduce how a word was *written* by recognizing its components in the left-to-right (or right-to-left) fashion.

Alternatively, there is a class of *perception-oriented* methods that perform a human-like *reading* by utilizing some stable features to reliably find

characters anywhere in the image, and to use them to bootstrap a few word candidates for a final evaluation phase. Such methods have rarely been truned into complete recognition systems. Instead, they are sometimes used to enhance more traditional “directional” algorithms described above.

To date, most attention appears to have been directed toward small- and limited-lexicon methods, especially those applied to Latin scripts. Many different methods and a few complete recognition systems have been proposed in the literature. Overall, however, the field of handwritten text recognition has not yet matured [14].

There are relatively few review-style articles [10, 4] and even fewer attempts to experimentally compare different methods. One reason for this is lack of sufficiently large, general, and publicly available test datasets. Furthermore, the performance of a complete ICR system depends on many other components besides its recognition engine, e.g. preprocessing, segmentation, and post-processing modules. To our knowledge, there have been virtually no attempts to create a common framework that allows an objective comparison of different engines or complete recognition systems.

One of the most successful approaches to Arabic character and word recognition is based on the application of *Hidden Markov Models* (HMM) [5, 9, 2]. They model variations in printing and cursive writing as an underlying probabilistic structure, which cannot be observed directly [1]. When small lexicons are involved, elementary HMMs that model individual characters are combined to form individual larger *model-discriminant* HMMs for every word in the lexicon. When a word image is presented to such a system for recognition, it chooses the word interpretation whose model gives the highest probability given a sequence of observations, or symbols, generated from the given image. For larger lexicons only one *path-discriminant* HMM is build from the character sub-HMMs and recognition is performed by finding the most likely path through the model given the sequence of observations from a word image. Path-discriminant models are more general but less accurate than model-discriminant ones.

HMM-based methods are popular because they performed quite well on a similar problem of speech recognition, and because their mathematical machinery has been developed and refined quite well. At the same time, they still constitute an active area of research. One reason is that performance of an HMM depends strongly on the type of features used to generate observation sequences both for training of the models and for recognition. As pointed out in [14], the “ideal” feature set has not yet been found, and may not exist at all. Also,

while most HMM-based methods represent a word image as a one-dimensional sequence of observation vectors, a more recent approach is to generalize to two-dimensional Markov Models (also called Markov Random Fields) [7]. While this approach appears to give better accuracy, it is also much more computationally expensive and so requires more sophisticated design and implementation than traditional HMM-based methods. In addition, MRF-based systems usually require much more training data than HMM-based ones.

Figure 2 shows the same Arabic word written by two different people (also taken from [8]). We can see that while the two handwriting styles are quite different, most of the large-scale features such as loops, turning points and long straight strokes are preserved. Therefore we made the decision to base feature extraction in our recognition system on such features. We believe that they are more robust than small-scale features such as local curvature and stroke thickness to inter-writer and intra-writer variations in size, orientation, and local distribution of characters as well as to poor image quality.

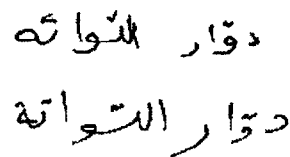


Figure 2: Same word written by different writers

Our approach, which we describe in more detail in Section 2 is based on that developed by Khorsheed in [5]. Our recognition system consists of four main components. *First*, it is based on large-scale geometric features. *Second*, it simulates the writing process by tracing connected components of a word image using a consistent set of rules. *Third*, it is trainable and uses HMMs. *Finally*, it can be adapted to use various types of high-level information at latter stages of the recognition process to resolve ambiguity.

2 Methodology

In Section 2.1, we present in some detail the original approach of Khorsheed [5]. We then describe our modifications in Section 2.2.

2.1 Original Algorithm

Khorsheed’s approach is to construct a single “universal” discrete HMM that represents an entire lexicon that is being modeled. This is an example

of the path-discriminant approach described in Section 1. The universal model is a collection of states each of which represents a single Arabic character. These character states are interconnected according to character bigram statistics extracted from a given training lexicon. Each state is in itself a discrete HMM that has a left-to-right (or, perhaps more precisely, right-to-left) topology. Each hidden state in a character model heuristically represents a stage in writing this character. Therefore the number of hidden states is proportional to the character’s complexity.

The recognition system works in two stages: training and recognition (see Figure 3). During training, a preprocessor converts an image of a word or a letter into a sequence of discrete observations symbols. These sequences are used to train individual character models that are then connected into a universal model. The well-known Baum-Welch HMM training algorithm [11] is used at this stage. During recognition, the same preprocessor converts a word image into an observation sequence that is then fed into the trained model. The most likely path through the model yields a sequence of character states that constitutes the recognized word. A modification of the standard Viterbi algorithm [11] that computes several top paths rather than just the best one is used at this stage.

The preprocessor works in several steps. First, a skeleton of a word or letter image is computed and represented as an image consisting of background pixels, “simple” foreground pixels as well as “knot” points that are foreground pixels that are end or intersection points. Then this image is converted into an ordered sequence of links that connect the knots. This approach is somewhat similar to the method described in [15] where knots are used as “anchors” to align both topological and geometric features of a glyph skeleton. In order to perform the skeleton traversal, we use a set of tracing rules that simulate the writing process. However, it is important to realize that *any* set of rules would work as long as it is consistent and robust to variations. For instance, the main right-to-left direction of tracing is not essential. After the links are computed, the next step is to detect loops of various types. After that all links that are not part of a loop are approximated by piecewise-linear curves. The final step of the preprocessor is to convert the resulting sequence of line segments into a sequence of observations. Each segment is either labeled as a part of a large-scale geometric feature such as a loop, an intersection point or a turning point, or quantized using k -means clustering according to its orientation and length. The label or the index of the k -means cluster centroid is used to compute a discrete observation symbol.

A more detailed description of Khorsheed’s algorithm may be found in [5].

2.2 Modifications

We have made several modifications to the original methodology.

2.2.1 Smaller Feature Set

The complete feature set used in the original algorithm is shown in [5, Table 1]. In our preliminary work with synthetic data we reduced the number of features by computing fewer types of turning points and by reducing the number of k -means clusters (see Table 1). Experiments indicate that such a reduced set is sufficient to process the data at hand. We do realize, however, that as we get more involved with real-life data, the feature set may have to be expanded to provide more discriminative power. Such expansion will require relatively small changes to the recognition system as long as we continue to base it on discrete HMMs.

2.2.2 Separate Models for Various Character Forms

In the original algorithm, a given character model is used for all (2 or 4) positional forms of that character. We decided to use separate models for all the different form. The main reason for such a change is that, although this makes the resulting universal model several times bigger, it also makes it more accurate by limiting the number of connections between character models and therefore prohibiting or discouraging certain paths that would be allowed by the smaller “form-independent” universal model. For instance, there will be relatively few connections from models that represent final forms of characters into those representing initials character forms, because these can only occur in words consisting of two or more subwords, e.g. على با با (“Ali-Baba”).

2.2.3 Training by Words Without Segmentation

Although during the recognition stage, one does not need to pre-segment the data beyond separation into words, during training manual segmentation into characters is still required for the original algorithm. Thus training of Khorsheed’s system proceeds as follows.

1. For each training word:
 - 1.1 Manually pre-segment the word image into character sub-images.
 - 1.2 Convert all character sub-images into observation sequences.
2. For each character:

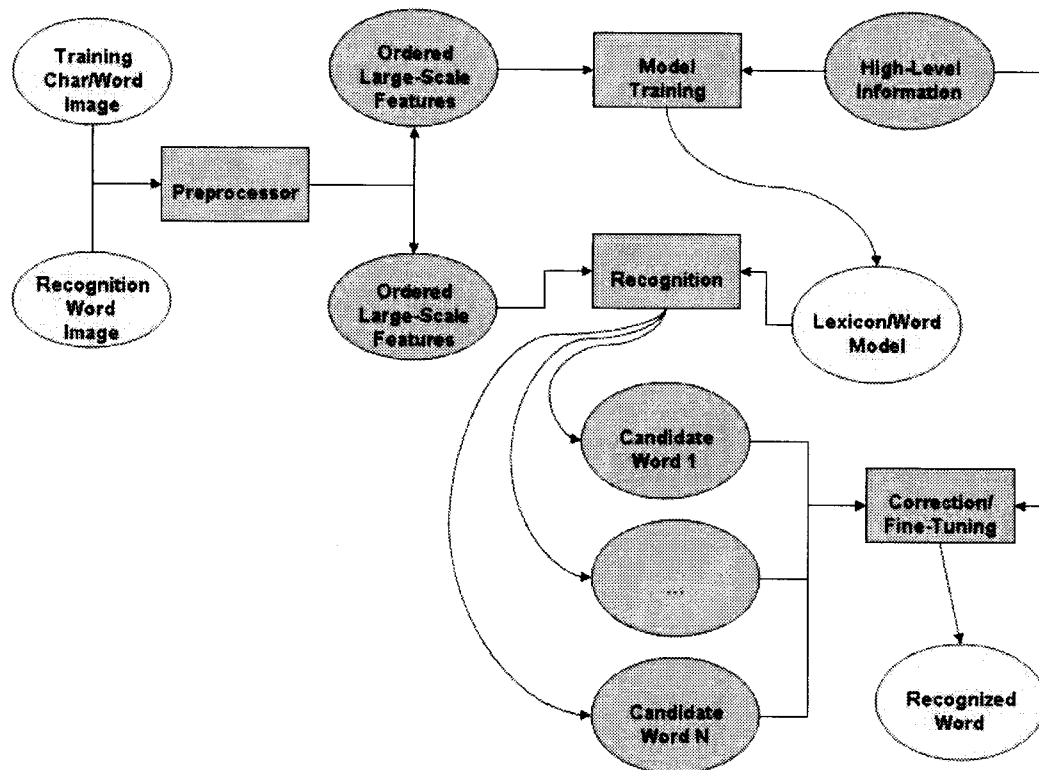


Figure 3: Block diagram of the recognition system

Observation Symbol	Feature Description	Observation Symbol	Feature Description
0	Dot	5	2-link complex loop
1	End point	6	3-link complex loop
2	Branch point	7	double loop
3	Cross point	8	Left turning point
4	Simple loop	9	Right turning point
		10-29	<i>k</i> -means cluster centroids

Table 1: Discrete features and the corresponding observation symbols

- 2.1 Compute the initial model. In particular, first determine the number of states.
 - 2.2 Collect all training sequences for that character from data generated in Step 1.
 - 2.3 Train the character model.
3. Connect all character models into a universal model using character bigram statistics.

Note that in Step 2 there is a single instance of each character model that is trained once using all instances of that character extracted from all training words. Manual preprocessing of data is a big limitation for a real-life recognition system. We propose a way of modifying this approach to avoid the pre-segmentation. We assemble a word model for each training word possibly using several identical copies of the initial character model if the corresponding character appears in that word more than once. For instance, the name علي بابا (“Ali-Baba”) contains two instances of the initial form of the letter ب (“Beh”). We then train the word models by corresponding observation sequences generated from the entire word images, just like when the model-discriminant approach to recognition described in Section 1 is used. Then we disassemble the trained word models into character components. Since the final universal model contains a single instance of each character model, after all training words have been processed, we have to combine all the instances of the same character models into one. This can be done by averaging. Thus, the training stage of the modified algorithm proceeds as follows.

1. For each character:
 - 1.1 Compute the initial model.
2. For each training word:
 - 2.1 Assemble initial character models into a word model.
 - 2.2 Convert entire training word image into an observation sequence.
 - 2.3 Use the sequence to train the word model.
 - 2.4 Disassemble the word model into instances of trained character models.
3. For each character:
 - 3.1 Average all instances of corresponding trained model into a single trained character model.
4. Connect all character models into a universal model using character bigram statistics.

In Section 3 we describe results of experiments with both manually segmented and unsegmented data.

3 Experiments

In this paper we present results of the initial “proof-of-concept” stage of our project that only involved experiments with synthetic data.

3.1 Manual Skeletonization and Tracing

In place of a fully automatic preprocessor we have implemented an interactive graphical tool designed to manually create ordered feature sequences of segmented letters. A user can superimpose an ordered piecewise-linear skeleton on top of an image of a “template” letter (see Figure 4). Each segment of the skeleton is labeled with a corresponding feature from Table 1 if this segment is a part of a loop, intersection etc. These skeletons are used to train individual character models and can also be concatenated into skeletons representing entire words. To simulate handwriting differences, both character and word skeletons can be randomly perturbed. Two types of perturbation are possible, namely,

- Adding random noise to geometric coordinates of endpoints of a segment (a.k.a. “shaking”). The amount of shaking is controlled by a standard deviation parameter.
- Random adding or removal of entire segments to/from a skeleton.

For instance, Figure 5 shows skeletons of the word محمد (“Mohammad”) created by concatenation followed by various degrees of shaking of the letter skeletons shown in Figure 4.

3.2 Synthetic Data Tests

We performed three sets of tests with synthetic data. In the first two cases, we used machine-printed Arabic characters as templates when manually creating character skeletons. The *first* set of tests was based on training using segmented data. To train a total of 102 character models for the various positional forms of the main 29 letters of the Arabic alphabet, we manually created and labeled skeletons of each character form. Each of these skeletons was perturbed by various degrees of shaking and this data was used to train individual models that were then assembled into a universal model based on bigram statistics extracted from a set of about 2,000 unique words collected from a set of newspaper articles. The character models contained between 3 and 20 hidden states. The same word collection was used to evaluate the trained model: word skeletons were obtained by concatenation of the corresponding character skeletons, then they were shaken.

The *second* set of tests involved unsegmented training data. In this case, 90% of the word collection was used for training and the other 10%

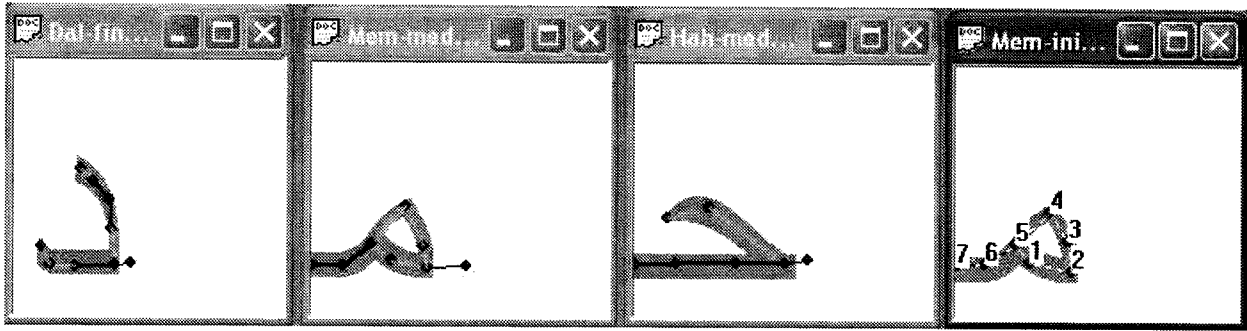


Figure 4: Manually created and labeled skeletons of letters م (“Mem”, initial and medial forms), ح (“Hah”, medial form) and د (“Dal”, final form)

for testing. For each training word, we created a word skeleton by character skeleton concatenation and generated an initial word model by using the same initial (untrained) character models as in the first set of tests. In this case, the word skeletons were perturbed both by shaking and by random addition and removal of segments. Then training was performed as described in Section 2.2.3. Test data was then created in a similar way for the test words and the universal model was evaluated. Performance of the recognition system for the two sets of tests is reported in Table 2. As expected, recognition accuracy for the segmentation-based tests was higher than for tests without segmentation. However, even in the latter case, it was high enough to warrant the use of this methodology in processing of more realistic data.

The *third* test involved several handwritten samples from the corpus described in [8]. We wanted to show that, at least using data that is mostly manually preprocessed, it is possible to model one handwriting style using perturbed versions of other styles. From the corpus, we selected two different words that, in particular, had three letters in common. For each word, we found three handwritten samples of the word from three different writers. In addition, we generated a machine-printed copy of each word. Using the interactive tool described in Section 3.1, we manually created and labeled skeletons for all eight samples (see Figure 6). Using the same methodology as in the second set of tests described above, we trained two word models using observation sequences obtained from perturbed versions of the skeletons of the handwritten samples. Then we disassembled the word models and created a universal model using bigram statistics from the two words and averaging instances of those character models that appeared more than once. We then tested the universal model using the observation sequences obtained from (unperturbed) skeletons of the machine-printed versions of the two words. In

both cases, our recognition system found the correct paths through the universal model. We also found that perturbation of training samples was critical: when we did not use shaking when training the word models, we could not find *any* path through the resulting universal model for one of the test observation sequences.

4 Future Work

This section explores various ways of extending our system, e.g. automating the initial image processing stage and the high-level information usage during the recognition stage.

4.1 Building Automatic Pre-processor

The current image preprocessing stage is largely manual. One of our immediate goals is to build a robust image preprocessor. We plan to use the algorithm by Zhang and Suen [16] for efficient thinning, followed by skeletonization via the curve fitting method by Liao and Huang [6]. Then after the proper segment labeling and ordering, the skeleton would be fed to the existing HMM-based recognition engine.

Although our system aims primarily at processing bitonal images, grayscale imagery can provide better recognition cues. Razdan et al. proposed an interesting approach to aid handwriting analysis and OCR [12, 13] by applying 3-dimensional modeling to analysis of handwritten documents. Their technique permits recovery of 3D spatial-temporal information from a 2D grayscale document image, potentially turning an off-line recognition problem into an on-line recognition problem that could be easier to solve. We could employ this technique in the pre-processing stage of our system to naturally order the sequence of HMM observations.

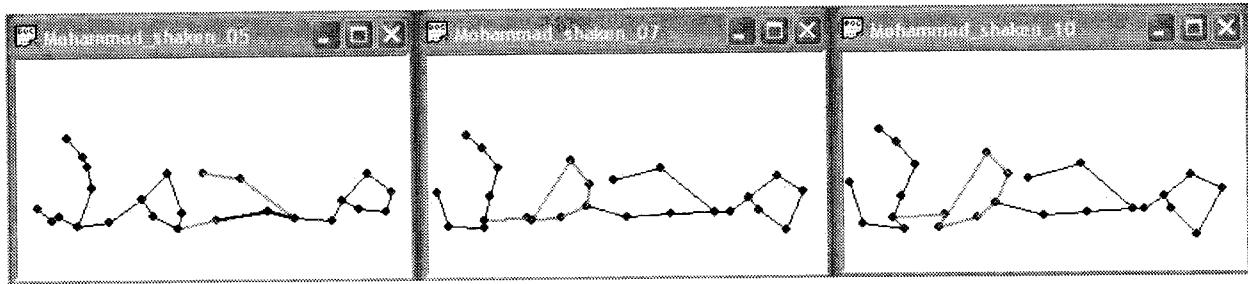


Figure 5: Shaken skeleton of the word محمد (“Mohammad”) obtained by concatenation of skeletons of the letters shown in Figure 4

4.2 Using High-level Information

Ambiguity is unavoidable in handwriting recognition. Figure 7 shows an example of a word image which may equally likely be interpreted as “dear” or “clear” even by the best recognition system - a human - unless additional context information is provided. Therefore we believe that incorporation of



Figure 7: Example of ambiguity

multiple types of high-level information into a handwriting recognition system will be essential for producing high quality results. This information may include limited lexicons, various language models (such as the letter and word n -grams), as well as context, location and co-location information.

References

- [1] Richard G. Casey and Eric Lecolinet. A survey of methods and strategies in character recognition. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 18(7):690–706, July 1996.
- [2] Michael Decerbo, Ehry MacRostie, and Premkumar Natarajan. The bbn byblos pashto ocr system. In *HDP '04: Proceedings of the 1st ACM workshop on Hardcopy document processing*, pages 29–32, New York, NY, USA, 2004. ACM Press.
- [3] Shimon Edelman, Tamar Flash, and Shimon Ullman. Reading cursive handwriting by alignment of letter prototypes. *International Journal of Computer Vision*, 5(3):303–331, 1990.
- [4] Mohammad Khorsheed. Off-line Arabic character recognition - a review. *Pattern Analysis and Applications*, 5(1):31–45, 2002.
- [5] Mohammad S. Khorsheed. Recognising handwritten Arabic manuscripts using a single hidden Markov model. *Pattern Recognition Letters*, 24(14):2235–2242, 2003.
- [6] C.-W. Liao and J. S. Huang. Stroke segmentation by Bernstein-Bezier curve fitting. *Pattern Recognition*, 23(5):475–484, 1990.
- [7] Hee-Seon Park and Seong-Whan Lee. A truly 2-d Hiddne Markov Model for off-line handwritten character recognition. *Pattern Recognition*, 31(12):1849–1864, 1998.
- [8] M. Pechwitz, S. Snoussi Maddouri, V. Märgner, N. Ellouze, and H. Amiri. IFN/ENIT-database of handwritten Arabic words. In *Proceedings of CIFED*, pages 129–136, 2002.
- [9] Mario Pechwitz and Volker Märgner. HMM based approach for handwritten Arabic word recognition using the IFN/ENIT-database. In *Proceedings of ICDAR*, 2003.
- [10] Rejean Plamondon and Sargur Srihari. Online and off-line handwriting recognition: a comprehensive survey. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 22(1):63–84, January 2000.
- [11] Lawrence R. Rabiner. A tutorial on HMM and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.
- [12] Anshuman Razdan, John Femiani, and Jeremy Rowe. 3D methods to aid handwriting analysis and OCR. In *Proceedings of the Symposium on Document Image Understanding Technology*, page 287, April 2003.
- [13] Anshuman Razdan, John Femiani, and Jeremy Rowe. 3d techniques for analyzing handwritten manuscripts for digital libraries. In *Inter-*

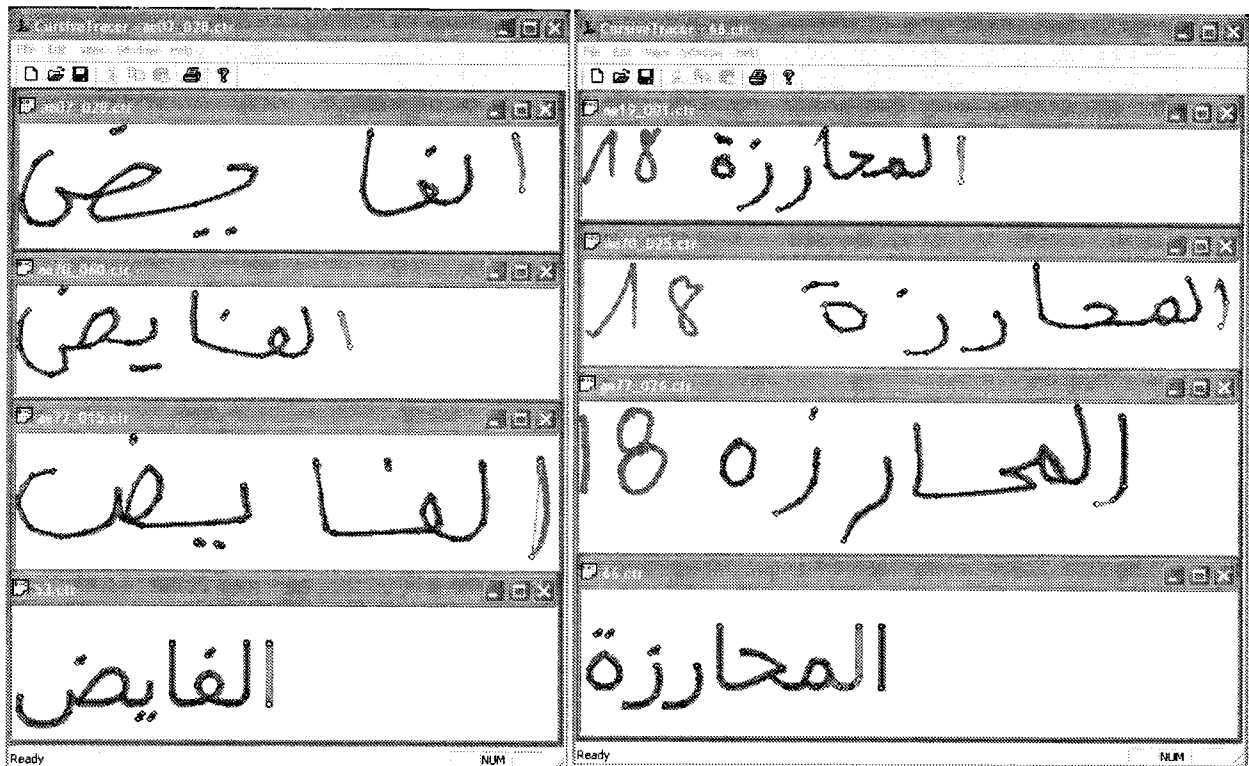


Figure 6: Two words, three handwritten and one machine-printed styles of each, manually preprocessed

national Conference on Digital Libraries, pages 430–436, New Delhi, February 24–27 2004.

- [14] Tal Steinherz, Ehud Rivlin, and Nathan Intrator. Off-line cursive script word recognition - a survey. *International Journal of Document Analysis and Recognition*, 2(2–3):90–110, 1999.
- [15] Mark A. Walch and Donald T. Gantz. Pictographic matching: a graph-based approach towards a language independent document exploitation platform. In *HDP '04: Proceedings of the 1st ACM workshop on Hardcopy document processing*, pages 53–62, New York, NY, USA, 2004. ACM Press.
- [16] T. Y. Zhang and C. Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984.

Training ...	Character Accuracy		Word Accuracy		
	Precision	Recall	Number of Word Seq's	Number of Matches	Recognition Rate
with Segmentation	99.5%	99.6%	107950	105691	97.9%
without Segmentation	93.1%	92.1%	10800	8059	74.6%

Table 2: Performance of the recognition system on synthetic data when training on both segmented and unsegmented data

Classification of Machine Print and Handwriting in Mixed Arabic Documents

Karthik Sridharan Faisal Farooq Venu Govindaraju

CEDAR, Suite 202, SUNY at Buffalo

Amherst, NY 14228, USA

{ks236,ffarooq2.govind}@cedar.buffalo.edu

Abstract

In this paper we present a system for classification of machine printed and handwritten text in mixed Arabic documents. The classification is performed at the word level. We propose a feature extraction algorithm for each word image based on Gabor filters followed by classification using an Expectation Maximization(EM) based probabilistic neural network. The main contributions of this paper are a Gabor filter based feature extraction for classification and design of an EM based neural network that reduces overfitting of training data. An overall precision of 94.62% was obtained using our method as compared to 83.33% using a simple backpropagation neural network and 90.26% using an SVM.

1 Introduction

The processing of document images prior to recognition plays a significant part in the development of Handwriting Recognition (HR) systems. In a document that has both machine print and handwritten text, it is important to distinguish between the two. This task is challenging in Arabic because the script is cursive in both machine print and handwriting and there are no special rules that can easily distinguish them.

In this paper we describe a method that extracts texture features from Arabic words. An EM based neural network is used for classification to deal with the sparse training data that does not have representatives from all fonts and writing styles.

A neural network based classifier was suggested in [1] that used nine texture features to distinguish machine print from the handwritten text in bank checks. Srihari et al. [2] describe a block separation method where the classification is based on the frequency of the heights of the different components in the segmented block. It is assumed that a block with widely differing heights is handwritten and a block with uniform component heights is machine printed. This method does not apply readily to Arabic script.

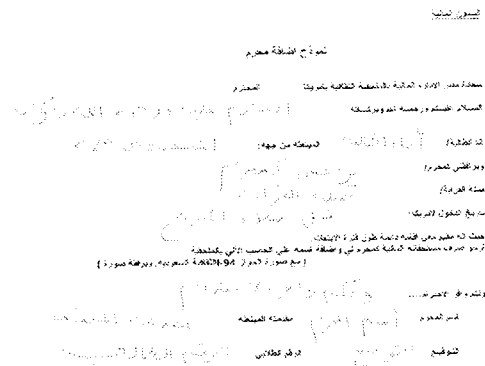


Figure 1: A sample document

Our hypothesis is that in Arabic handwriting, horizontal runs and gradients are not as uniform as in machine print. The advantage of our method is that it can be implemented at the word level as it captures the local structure of components in the document. A discrimination method that operates at the word level was described in [3], using slope and stroke width histograms. However, the method was not trainable and thresholds were selected empirically.

Figure 1 shows a block diagram of Our approach. It has 3 stages : (i) word extraction (ii) feature extraction and (iii) classification. In the word extraction stage, we binarize [7] the image and extract individual word images from the document [8]. Each word image is normalized by scaling to a fixed height while preserving the aspect ratio, hence the width of the word images vary. The baseline of the connected components is estimated and corrected. Lines are segmented into words by clustering connected components based on thresholds chosen by the normalized variance of white space in the lines. Directional Gabor filters are used to extract features from the word image. Classification is performed by a probabilistic neural network which is trained using an

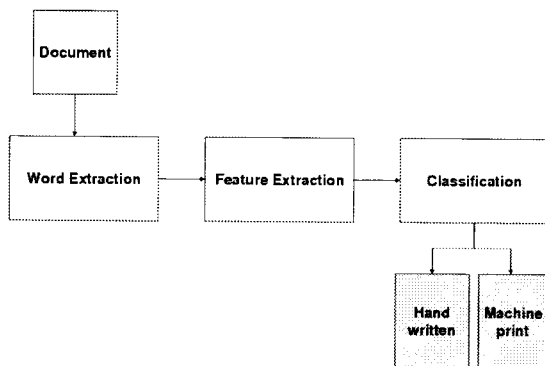


Figure 2: Components of the system

EM algorithm. This neural network combines solutions according to their posterior distribution to avoid overfitting based on the training data.

2 Feature Extraction

Gabor filters are directional filters that have been used successfully for classification of textures and automatic script identification [4]. Since direction of strokes and uniformity is a key feature in Arabic script, the use of Gabor filters seem to be ideally suited for the task.

Gabor functions are Gaussian functions modulated by a complex sinusoid. In $2D$, a Gabor function is given by:

$$h(x, y) = g(x', y') \cdot e^{2\pi j F x'}$$

$$g(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{1}{2}[(\frac{x}{\sigma_x})^2 + (\frac{y}{\sigma_y})^2]}$$

where (x', y') are rotated components of (x, y) ,

$$x' = x \cos\theta + y \sin\theta$$

$$y' = -x \sin\theta + y \cos\theta$$

and F is the radial frequency which for a given scale s is given by $F = F_0/s$. The output of the filter,

$$G_{\theta,s}(x, y) = \int I(s, t) h_{\theta,s}(x - s, y - t) ds dt$$

is an image with the components in the chosen direction becoming prominent. Since machine print Arabic has more uniformity as compared to handwriting and the same characters repeated in the text have strokes in the same direction, Gabor filters for feature extraction is a prudent choice. Figure 2 shows a sample word image extracted. Figure 3 shows the output of the Gabor filter for each direction at a single scale when applied to the word image in Figure 2.



Figure 3: Word image

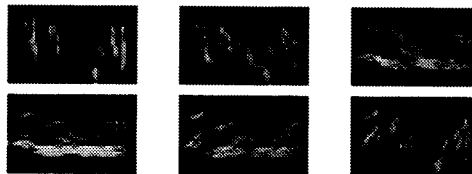


Figure 4: Gabor Filter Output for 1 scale and 6 orientations

Since the word images all vary in their width the Gabor filter cannot be applied directly. For classifiers like neural networks or support vector machines (SVM) the feature vectors need to be of fixed size. This problem can be resolved by noting that the main information obtained from the Gabor filter output is the strength of the word image in each direction and scale which is given by the sum of the output of the filter for each direction and scale resulting in a vector of size [number of scales \times number of directions]. In order to make it font independent we normalize the output by dividing the sum of filter output by the sum of the output of an isotropic Gaussian filter,

$$Gauss(x, y) = \int I(s, t) g(x - s, y - t) ds dt$$

For direction θ and scale s

$$Gabor(\theta, s) = \frac{\int G_{\theta,s}(x, y) dx dy}{\int Gauss(x, y) dx dy}$$

In our implementation we use a set of 6 filter banks with 3 directions and 2 scales. Thus for each word image we extract a 6-dimensional feature vector for classification.

3 Classification

The training set is generally sparse and does not cover all fonts. Traditional classifiers like SVMs and backpropagation neural networks tend to overfit sparse data. This drawback is overcome by the Bayesian Neural Networks (BNN) [5] by integrating over the posterior distribution of the weights. That is, instead of finding one solution, many solutions are found and are weighted according to their posterior probabilities given the training data. The BNN outperforms many classifiers including the SVM. However BNNs need to sample high dimensional weight vectors from their posterior distribution. Markov Chain Monte-Carlo sampling methods, such as Langevin Monte Carlo method and

Hamiltonian sampling methods can be used for the purpose. However these methods are computationally expensive.

A BNN for a binary classification can be viewed as linear combination of potential solutions according to their posterior probabilities. Since sampling is computationally intensive, we propose a new neural network where a layer of neurons use an error function which apart from penalizing neurons responsible for errors in classification, also penalizes neurons that are similar to each other.

The idea is to make the neurons compete in finding different possible solutions. The part of the error function that penalizes solutions that lead to bad classification is given by the Euclidean distance between the target and the output. The part of the error function that penalizes similar solutions is given by the sum of the square of the cosine of the neuron weight vector with respect to the weight vectors of the other neurons in the layer. A bias term is included in the weight vector to make sure that all the hyperplanes given by the neurons need not pass through the origin. Thus the error function of a single neuron is given by

$$E_{i,k} = \frac{1}{2}(t_k - o_k)^2 + \frac{\beta}{2} \sum_{j \neq i} \frac{w_i^T w_j}{|w_i| |w_j|}^2 \quad (1)$$

where

$$\cos(v, u) = \frac{v^T u}{|v| |u|}$$

and t_k is the target for the k^{th} instance and o_k is the weighted sum of the output of all the neurons according to their posterior probabilities. Therefore,

$$o_k = \sum_i P(w_i) o_{k,i}$$

The transfer function used is the classic sigmoid function. One way of looking at this error function is as the negative log likelihood of the posterior. In this view we would be modeling the likelihood of the output to follow a Gaussian distribution with mean around the target and the prior to be a Gaussian distribution of the cosine function of the weight of the neuron with the other neurons with zero mean. Zero mean of the cosine signifies that we are trying to model the neurons such that the cosine of their weights with the other neurons is as orthogonal as possible since $\cos(90) = 0$. Parameter β decides the trade off between the error on classification and how "different" the solutions should be. By minimizing the error function we obtain weights that are as orthogonal (different) to each other as possible and yet classify well.

We also need to weight the solutions given by the neurons according to their posterior probabilities.

We propose the use of an Expectation Maximization (EM) algorithm [6] for learning the weights and probabilities. In the E step of the algorithm we find the posterior probabilities of the neurons given the training data. In the M step we optimize the weight parameters to reduce error. While the E step is exact, the M step uses the gradient descent approximation. If we take the error function to be negative log likelihood then in probability space for the k^{th} instance and i^{th} neuron we get posterior probability of the weight as

$$\begin{aligned} & P(w_i | o_k = t_k, w_{j:j \neq i}, I_k) \\ &= \frac{P(o_k = t_k | w_i, w_{j:j \neq i}, I_k) P(w_i | w_{j:j \neq i}, I_k)}{\sum_i P(o_k = t_k | w_i, w_{j:j \neq i}, I_k) P(w_i | w_{j:j \neq i}, I_k)} \\ &\propto \epsilon^{-\frac{\eta}{2}(o_k - t_k)^2} \epsilon^{-\frac{\beta \eta}{2} \sum_{j \neq i} \frac{w_i^T w_j}{|w_i| |w_j|}^2} \\ &\propto \epsilon^{-E_{i,k}} \end{aligned}$$

where η is the learning rate for the gradient descent on weights and β is a free parameter. I_k denotes the k^{th} input. The EM algorithm uses a batch update of weights so all the instances are considered at once and the parameters of all the neurons are updated simultaneously. Let w_i^t represent the weight of the i^{th} neuron after t updates by the algorithm. The probability of getting all the classifications correct given that the weights of the neurons in the previous step $t-1$ is W^{t-1} ($W^{t-1} = \{w_i^{t-1} | i = 1 : n\}$ where n is the number of neurons) is

$$\begin{aligned} & \prod_k P(o_k = t_k | W^{t-1}, I_k) \\ &= \prod_k \sum_i P(o_k = t_k, h_i^t | W^{t-1}, I_k) \end{aligned}$$

and h_i is a hidden variable.

We assume that the instances are independent. Taking log on both sides,

$$\begin{aligned} & \sum_k \log(P(o_k = t_k | W^{t-1}, I_k)) \\ &= \sum_k \log\left(\sum_i P(o_k = t_k, h_i^t | W^{t-1}, I_k)\right) \end{aligned}$$

Now we simply multiply and divide the term in the sum by some positive non-zero $\alpha_{i,k}^t$ where $\sum_i \alpha_{i,k}^t = 1$ (The α 's are probability distribution of the latent or hidden variables) and hence,

$$\begin{aligned} & \sum_k \log(P(o_k = t_k | I_k)) \\ &= \sum_k \log\left(\sum_i \frac{P(o_k = t_k, h_i^t | W^{t-1}, I_k) \alpha_{i,k}^t}{\alpha_{i,k}^t}\right) \end{aligned}$$

By Jensen's inequality we have

$$\begin{aligned} & \sum_k \log(P(o_k = t_k | I_k)) \\ & \geq \sum_k \sum_i \alpha_{i,k}^t \log\left(\frac{P(o_k = t_k, h_i^t | W^{t-1}, I_k)}{\alpha_{i,k}^t}\right) \end{aligned}$$

Further splitting the joint probability $P(o_k = t_k, h_i^t | W^{t-1}, I_k)$ as the product of $P(o_k = t_k | W^{t-1}, I_k)$ and $P(h_i^t | o_k = t_k, W^{t-1}, I_k)$ we get,

$$\begin{aligned} & \sum_k \log(P(o_k = t_k | I_k)) \\ & \geq \sum_k \sum_i \left\{ \alpha_{i,k}^t \log\left(\frac{P(o_k = t_k | W^{t-1}, I_k)}{\alpha_{i,k}^t}\right) \right. \\ & \quad \left. + \log\left(\frac{P(h_i^t | o_k = t_k, W^{t-1}, I_k)}{\alpha_{i,k}^t}\right) \right\} \end{aligned}$$

Let $O = T$ represent the event when all instances are classified correctly and let

$$\begin{aligned} F(\alpha^t, W^{t-1}) &= \\ & \sum_k \sum_i \left\{ \alpha_{i,k}^t \log\left(\frac{P(o_k = t_k | W^{t-1}, I_k)}{\alpha_{i,k}^t}\right) \right. \\ & \quad \left. + \log\left(\frac{P(h_i^t | o_k = t_k, W^{t-1}, I_k)}{\alpha_{i,k}^t}\right) \right\} \quad (2) \end{aligned}$$

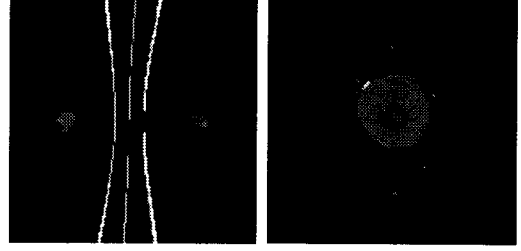
Therefore we have

$$\log(P(O = T | I)) \geq F(\alpha^t, W^{t-1})$$

Since we cannot build an algorithm that directly optimizes $\log(P(O = T | I_k))$ we instead optimize $F(\alpha^t, W^t)$ as this guarantees that the likelihood always increases. Further the expectation maximization algorithm alternately optimizes the α 's and the weights keeping one constant while optimizing the other thus performing a co-ordinate ascent. In the E step the α 's are optimized and in the M step the weights are optimized and thus the EM algorithm iterates between successive E and M steps.

In the E step we need to minimize $F(\alpha^t, W^t)$ with respect to α^t . Hence we differentiate $F(\alpha^t, W^t)$ with respect to each $\alpha_{i,k}^t$ keeping weights constant and solve the equation for the differential equal to 0. However it is important to note that since α 's are 0, they should sum to one for each instance. Hence we use the Lagrange multiplier λ in the equation and get

$$\begin{aligned} & \frac{dF(\alpha^t, W^{t-1})}{d\alpha_{i,k}^t} = \\ & \log\left(\frac{P(o_k = t_k | W^{t-1}, I_k) P(h_i^t | W^{t-1}, I_k)}{\alpha_{i,k}^t}\right) - 1 + \lambda \end{aligned}$$



(a) Linear Classification (b) Non-linear Classification

Figure 5: 2D classification Results

$$= 0$$

Therefore to minimize $F(\alpha^t, W^{t-1})$ with respect to the α 's, we have

$$\alpha_{i,k}^t = P(o_k = t_k | W^{t-1}, I_k) P(h_i^t | W^{t-1}, I_k) \epsilon^{\lambda-1}$$

But sum of α_k^t 's is equal to 1 for each data instance, thus

$$\epsilon^{\lambda-1} \sum_i P(h_i^t | W^{t-1}, I_k) P(o_k = t_k | W^{t-1}, I_k) = 1$$

To minimize $F(\alpha^t, W^{t-1})$ with respect to the α 's we simply set

$$\alpha_{i,k}^t = \frac{P(o_k = t_k | W^{t-1}, I_k) P(h_i^t | W^{t-1}, I_k)}{\sum_j P(o_k = t_k | W^{t-1}, I_k) P(h_j^t | W^{t-1}, I_k)}$$

Since the error function is the negative log of probability we have ,

$$P(o_k = t_k | W^{t-1}, \pi^{t-1}, I_k) \propto e^{-E_{i,k}}$$

where $E_{i,k}$ is got from Equation 3.

Now let

$$\pi_i^t = P(h_i^t | W^{t-1}, I_k)$$

be the prior associated with each neuron. Therefore, the update of α 's in the E step to minimize $F(\alpha^t, W^{t-1})$ is given by

$$\alpha_{i,k}^t = \frac{\epsilon^{-E_{i,k}} \pi_i^{t-1}}{\sum_j \epsilon^{-E_{j,k}} \pi_j^{t-1}} \quad (3)$$

From 2 we have

$$F(\alpha^t, W^t) = \sum_k \sum_i \alpha_{i,k}^t \log\left(\frac{P(o_k = t_k | W^{t-1}, I_k) \pi_i}{\alpha_{i,k}^t}\right)$$

In the M step we minimize $F(\alpha^t, W^t)$ with respect to each of the weights W^t and the priors. To find the optimal priors we differentiate $F(\alpha^t, W^t)$ with respect to each of the π 's. However, since the priors

Table 1: Performance analysis of the system.

	Back-Prop. Neural Net		SVM		EM Neural Net	
	Precision(%)	Recall(%)	Precision(%)	Recall(%)	Precision(%)	Recall(%)
Handwritten	62.26	95.19	74.26	97.12	94.68	85.58
Machine-print	97.83	79.02	98.82	87.76	94.93	98.25
Overall Performance(%)	83.33		90.26		94.62	

must add up to 1, we use the Lagrange multiplier λ' to ascertain this condition. Thus for minimizing $F(\alpha^t, W^t)$ with respect to the prior,

$$\begin{aligned} & \frac{dF(\alpha^t, W^t)}{d\pi_i^t} \\ &= \sum_k \frac{d(\alpha_{i,k}^t \log(P(o_k = t_k | W^{t-1}, I_k) \pi_i^t))}{d\pi_i^t} + \lambda' = 0 \\ &\Rightarrow \sum_k \frac{\alpha_{i,k}^t}{\pi_i^t} + \lambda' = 0; \end{aligned}$$

Therefore,

$$\pi_i^t = -\frac{\sum_k \alpha_{i,k}^t}{\lambda'}$$

But the priors sum to one. Therefore

$$\lambda' = -\sum_i \sum_k \alpha_{i,k}^t$$

For minimizing $F(\alpha^t, W^t)$ we set priors as

$$\begin{aligned} \pi_i^t &= \frac{\sum_k \alpha_{i,k}^t}{\sum_i \sum_k \alpha_{i,k}^t} \\ \Rightarrow \pi_i^t &= \frac{\sum_k \alpha_{i,k}^t}{K} \end{aligned} \quad (4)$$

Where K is the number of instances of training data.

To minimize $F(\alpha^t, W^t)$ with respect to each of the weights W^t we use the gradient descent. The gradient of $F(\alpha^t, W^t)$ with respect to each of the weights w_i^t is given by

$$\begin{aligned} \frac{dF(\alpha^t, W^t)}{dw_i^t} &\propto \sum_k \frac{d(\alpha_{i,k}^t \log(e^{-E_{i,k}} \pi_i^t))}{dw_i^t} \\ &\propto -\sum_k \alpha_{i,k}^t \frac{dE_{i,k}}{dw_i^t} \end{aligned}$$

Since $\alpha_{i,k}^t$ is constant for all i and k, we need to perform gradient descent on error function $E_{i,k}$. This means we simply follow the negative gradient of error function $E_{i,k}$ which is given by

$$-\frac{dE_i}{dw_i^{t-1}}$$

$$\begin{aligned} & \propto -\eta \sum_k (t_k - o_k)(1 - o_{k,i}) o_{k,i} I_k \\ & -\beta \eta \sum_{j \neq i} \frac{(w_i^{t-1})^T w_j^{t-1} w_j^{t-1} - w_j^{t-1} \frac{w_i^{t-1} (w_i^{t-1})^T}{|w_i^{t-1}|^2}}{|w_i^{t-1}| |w_j^{t-1}|} \end{aligned} \quad (5)$$

Where $o_{k,i}$ represents the output of the i^{th} neuron. Since we follow the negative gradient, the M step is partial. The EM algorithm is initialized with setting weights to some random vectors and making all priors equal. In the E step we use Equation 3 to update the posteriors based on the weights and priors from the previous iteration. In the M step we use Equation 4 to set the priors based on the calculated posterior and use the gradient given by Equation 5 to perform gradient descent on the weights of the neural network.

To initially test the performance of the neural network, we used some 2D test cases. Figure 5(a) shows the performance of the method on the linear classification problem. The central gray band stands for 0.5 probability region and the side gray bands stand for 0.75 and 0.25 probability regions respectively. The results obtained are similar to the Bayesian decision boundary. It curves out at places where data is not available thus showing that the system is unsure at places where training data is unavailable. Figure 5(b) shows the decision boundary found by the method on a non-linear test set. Here we see the gray band around the data points in the center which is the decision boundary which stands for 0.5 probability region.

4 Performance Analysis

There is a lack of labeled handwritten datasets for training and testing purposes in Arabic. We have collected handwriting samples from forms that have prompts in machine print. Figure 1 shows an example of the document. We collected 34 documents from 18 different writers. We used 5 documents for training purpose and the remaining for testing.

We measured the performance of our system by the precision and recall metrics, commonly used by the Information Retrieval (IR) community. Precision in our case would be the ratio of handwritten

words labeled correctly to all words that are labeled as handwritten by our system. Recall is measured as the ratio of handwritten words labeled correctly to all handwritten words in the test set. Similarly the corresponding metrics for machine print are also calculated. We used a training data of just 50 words each of handwritten and machine print images and a testing data of documents containing a total of 286 machine print and 104 handwritten words in them. Table 1 shows the summary of our experimental results. In order to evaluate the performance of our classification step, we compared the results by using a back-propagation neural network and an SVM for classification. The overall precision of our system is 94.62%. Our system outperformed a backpropagation neural network (83.33%) and also an SVM (90.26%).

5 Conclusion

Discrimination of handwritten and machine printed text is required in many document analysis and forensic applications. We have presented an algorithm for discriminating Arabic handwriting from machine print, which is a difficult task when compared to other languages. However, our method is trainable and relies on the uniformity of strokes and curves in machine print compared to handwriting. Given the training data, our method can be adapted to other languages and scripts as well. We compared our proposed classification technique with commonly used back-propagation neural network and state-of-the-art SVM. Our EM based neural network outperformed the standard systems. The EM based neural network addresses the problem of sparse data, thus our method is robust even when large amounts of training data is not available.

References

- [1] E.B.D.S. Jose, B. Dubuisson and F. Bortolozzi, Distinguishing between Handwritten and Machine Printed Text in Bank Cheque Images, *Document Analysis Systems, LNC'S* (2002) 2423, 58-61.
- [2] P.W. Palumbo, S.N. Srihari, J. Soh, R. Sridhar and V. Demjanenko, Postal Address Block Location in Real-Time, *IEEE Computer* (1992), 34-42.
- [3] F. Farooq, V. Govindaraju, and M. Perrone. Processing of Handwritten Arabic Documents, *Proceedings of the 12th Conference of the International Graphonomics Society* (2005) 183-186, Salerno, Italy
- [4] P.B. Pati, S.S. Raju, N. Pati and A.G. Ramakrishnan, Gabor filters for Document analysis in Indian Bilingual Documents, *Proceedings of International Conference on Intelligent Sensing and Information Processing*, (2004) 123-126, Chennai, India.
- [5] R.M. Neal, Bayesian Learning for Neural Networks, *Springer Verlag* (1996), New York.
- [6] Dempster A., Laird N., and Rubin D. *Maximum Likelihood from Incomplete Data via the EM Algorithm*, *Journal of the Royal Statistical Society, Series B*, 39(1):138, 1977.
- [7] N. Otsu, A threshold selection method from gray-level histograms, *IEEE Transactions on Systems Man and Cybernetics* (1979) 9(1), 62-66.
- [8] F. Farooq, V. Govindaraju, and M. Perrone. Pre-Processing Methods for Arabic Handwritten Documents, *Proceedings of the International Conference on Document Analysis and Recognition* (2005) 267-271, Seoul, Korea

Extraction and Information Retrieval

Accurate Document Categorization of OCR Generated Text

Robert Price¹ Anthony Zukas²

¹Content Analyst LLC, Reston, VA

²SAIC, Reston, VA

Abstract

The purpose of this research was to examine the application of the Latent Semantic Indexing (LSI) algorithm to the categorization of Optical Character Recognition (OCR) generated documents (or text). LSI is a robust dimensionality reduction technique for the processing of textual data. The technique can be applied to collections of documents independent of subject matter or language. Given a collection of documents, LSI indexing can be employed to create a vector space in which both the documents and their constituent terms can be represented. In practice, vector spaces of several hundred dimensions are typically employed. The resulting vector space possesses some unique properties that make it well suited to a range of information-processing problems. Of particular interest to the document conversion community is the fact that the technique is highly resistant to noise in text that is generated by the OCR conversion process. Noise in OCR generated documents nominally takes the form of missing characters or improperly interpreted characters that result in word misspellings. Normally, human operators are employed to perform corrective post processing of noisy OCR-generated text prior to categorization or other document workflow processes that require highly accurate text (e.g., Boolean searches). A technology such as LSI that could eliminate human review and editing of OCR-generated text, while maintaining highly accurate categorization or information retrieval rates, would result in a dramatic increase in workflow throughput with substantial labor savings.

1 Introduction

Previous work [1] has demonstrated the high performance of Latent Semantic Indexing (LSI) [5] on the Reuters-21578 [9] test set. In this paper we have examined the ability of LSI to categorize documents that contain corrupted or noisy text (e.g., misspellings, transliterations differences, OCR errors). In an earlier case study for a US Intelligence Agency related to a pending FOIA release, utilizing documents derived via OCR, indicated the LSI technology could possibly provide good retrieval performance on OCR generated text. Attempts at using other technologies to examine

the document set for potential “mosaic effect” connections had failed.

A key feature of LSI that makes it attractive for categorizing noisy text is its capability of automatically extracting the conceptual content of text items [5, 6]. 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 (or documents with noisy text) 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 (or variants thereof generated by noise sources, e.g., OCR). Of particular note is that the LSI algorithm does not employ any auxiliary data structures as part of its processing (e.g., thesauri, grammars, taxonomies or ontologies).

2 Discussion

Numerous pattern matching technologies have been studied and applied towards finding relevant search patterns in noisy text [2,4]. If the image conversion process is poor (i.e., originals are of poor quality) and the characteristics of the OCR engine are well understood a model of the overall conversion process can be created and employed to train a pattern matching engine for searching the OCR text. Such models are too expensive and time consuming to produce in practice. If the image conversion process is relatively clean (i.e., originals are of good quality, image enhancement software and production quality scanning equipment is used) then pattern matching technologies may be able to overcome some inconsistencies in the OCR text and achieve some reasonable confidence in the precision/recall of the search results. Often the real world is somewhere in between [11].

This problem is also very akin to the conversion of text from speech and subsequent retrieval of the text by search [3]. Techniques described in this paper are also applicable to this domain.

3 Approach

To formally examine the robustness of the LSI algorithm under conditions of noisy text a

categorization-style testing approach was selected since the authors had previous experience testing the LSI algorithm using this mode of operation. The Reuters categorization test sets were selected due to their availability and active use in other categorization testing efforts by other researchers. The Reuters categorization test sets come in two versions (Reuters-21578 [9] and Reuters RCV1-V2 [10]) and consist of document training and document test sets. The training set provides a knowledge base to train the algorithm undergoing examination and the test documents are used in evaluating the categorization accuracy of the subject algorithm. In the case of the LSI algorithm the training documents were employed to train the LSI vector space. The makeup of the Reuter's corpuses is discussed in the following section.

In the case of LSI the training documents are used to create a matrix that relates the documents and the words that occur in them. The rows of the matrix correspond to *terms* that occur in the documents. The columns correspond to individual *documents*. The number entered in the *i*th row and *j*th column of the matrix corresponds to the number of times that the *i*th term appears in the *j*th document. The matrix produced in this manner can be very large. In practical applications it can involve hundreds of thousands of terms and even larger numbers of documents. Fortunately, the matrix is very sparse and is amenable to dimensionality reduction.

A powerful mathematical technique, known as singular value decomposition (SVD), is used to reduce this matrix to a product of three matrices. One of these matrices has non-zero values only on the diagonal. Small values on this diagonal, and their corresponding rows and columns in the other two matrices are then deleted. This truncation process generates a matrix of greatly reduced dimensionality. For any given dimensionality, this technique can be shown to produce an optimal approximation of the original matrix. The columns of the associated matrices can be used to create a vector space in which both terms and documents are represented. The dimensionality of this vector space can be chosen to work well in a particular application. Typically, LSI spaces with a dimensionality of several hundred are employed. The dimensionality reduction has the effect of extracting semantic information that is latent in the processed text, hence the name latent semantic indexing. In some cases unlabeled background data is used to enhance the knowledge base of the training set [7]. This approach was not utilized in this work.

To test the robustness of LSI in conditions of increasing document degradation a procedure was developed that degraded percentages of text in the test documents by inserting, deleting, and substituting characters randomly at specified error rates based on experience with numerous OCR engines and OCR

degradation models published in the literature [4]. In our model, for any given document, 66 percent of the errors were random alphanumeric substitutions, 17 percent were deletions, 12 percent were insertions of random alphanumeric characters and 5 percent were random insertions of spaces. Although true OCR errors are more systematic, the intent here was to show to what extent the text of the documents could be degraded and still retain useful categorization results. Appendix A illustrates the impact of the degradation algorithm on one test document at various stages of degradation.

The collections of degraded test documents were then presented to the LSI algorithm for categorization and the accuracy results recorded.

4 Test Corpus and Performance Measures

To understand the performance of LSI on degraded text, the ModApte version of the Reuters-21578 test set [9] was utilized. A second test using the larger more recent RCV1-V2 test set was also performed to examine the impacts of a larger training set and much larger test set.

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.

The larger Reuters RCV1-V2 [10] consists of 804,414 documents broken up into 23,149 training documents and 781,265 test documents. We employed the Topic category orientation of the test set which consists of 103 Topic categories.

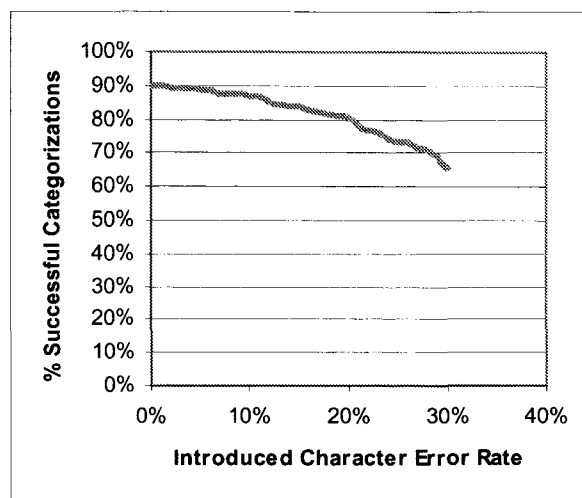


Figure 1. Reuters-21578 document categorization accuracy versus introduced character error rate.

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 [12], and the total ('micro-averaged') precision P and recall R (defined in [13]).

5 Results

The categorization accuracy of LSI on the Reuters-21578 test set was reported in detail in [1]. Figure 1 shows the performance of the LSI algorithm on the Reuters-21578 test set with regards to categorization accuracy for various percentages of document degradation. As can be observed in Figure 1 the categorization accuracy falls off at a very slow rate for an introduced character error rate of 0 to 30 percent.

To compare results across the two Reuters data sets the authors plotted baseline categorization accuracy versus the degradation levels of the degraded text. Figure 2 (Reuters-21578) and Figure 3 (Reuters RCV1-V2) show the plots of the overall categorization accuracy compared to the baseline versus the introduced character error rate.

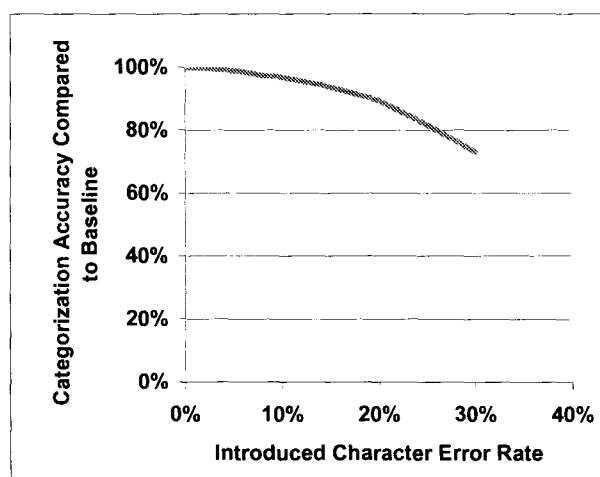


Figure 2. Reuters-21578 categorization accuracy compared to baseline versus introduced character error rate.

It is interesting to note that the two curves are very similar up to an introduced character error rate of 15 percent. Examination of Figure 3 revealed an interesting artifact in the Reuters RCV1-V2 curve. After an introduced character error rate of 15 percent, the curve for the Reuters RCV1-V2 test set falls off much faster than the curve for the Reuters-21578 test set. As noted previously the Reuters RCV1-V2 test set contains 718,265 test documents versus 2,581 test documents for the Reuters-21578 test set. One explanation for the differences (and of continuing author investigation) is the larger number of new unique word combinations generated by degrading the RCV1-V2. There is a 33:1 ratio for the number of test

documents versus training documents in the Reuters RCV1-V2 document set. This same ratio is 2.5:1 for the Reuters-21578 document collection (ModApte variation). It could be possible that a larger training set (than the one supplied with the RCV1-V2 test collection) is required at higher levels of document degradation.

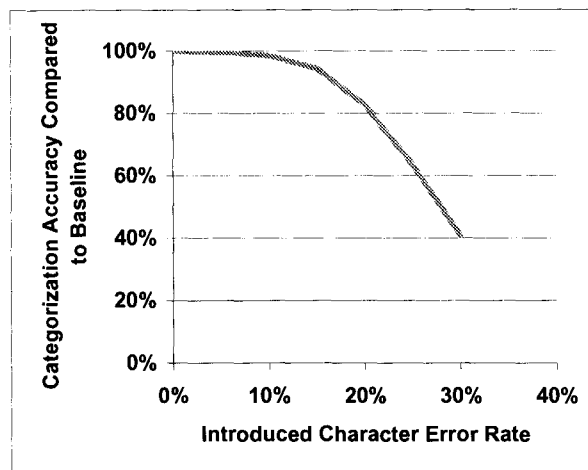


Figure 3. Reuters RCV1-V2 categorization accuracy compared to baseline versus introduced character error rate.

Figures 4 and 5 show plots of the baseline categorization accuracy versus the percentage of corrupted words. The interest in these plots stemmed from the observation that at a 20% introduced character error rate roughly three-quarters of the words in a test document are corrupted. It can be observed that the LSI algorithm maintains a very flat curve out to a point where approximately 60% of the words are corrupted in a test document.

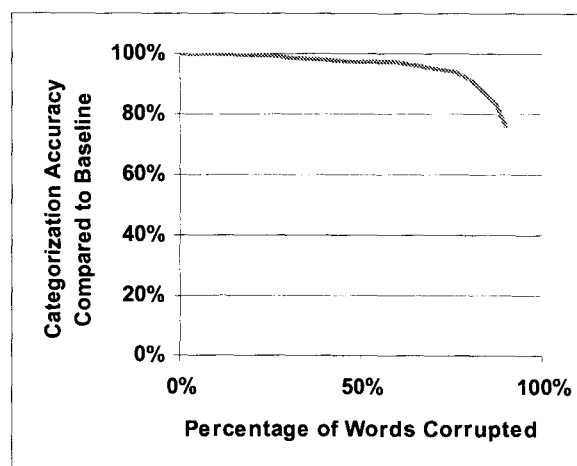


Figure 4. Reuters-21578 categorization accuracy compared to baseline versus the percentage of words corrupted.

To the authors' knowledge this level of information retrieval on noisy text has never been achieved using

other technologies and is worthy of further investigation. Further examination of Figures 4 and 5 also show the more rapid falloff at the tail of the curve between the two document collections as observed in Figures 2 and 3.

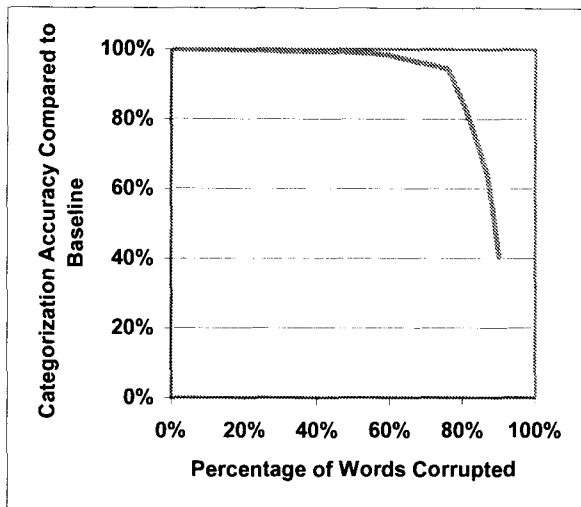


Figure 5. Reuters RCV1-V2 categorization accuracy compared to baseline versus the percentage of words corrupted.

6 Conclusions

The authors find these results very encouraging. The evidence that the accuracy of the LSI algorithm falls off very slowly in categorization testing, even at high levels of text errors, indicates LSI could be an excellent solution to the general indexing of noisy text. In most applications where OCR text is targeted for information retrieval, the text conversion process most often requires a human-in-the-loop to either (1) extract document meta data which is indexed (rather than the OCR text) or (2) reviewers read the OCR text and provide human-intensive review and correction prior to full-text indexing. The results of this study indicate that reasonably accurate OCR conversion processes combined with LSI indexing could eliminate the human review process normally associated with the indexing of OCR text.

Future research will address the application of LSI in other areas, e.g., indexing of machine readable text as a result of speech-to-text conversion, the use of LSI-based text summarization of noisy text to replace human-generated meta data, and the potential of using LSI to mitigate the effects of noisy data on machine translation systems.

Acknowledgements

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

References

- [1] A. Zukas and R. Price, Document Categorization Using Latent Semantic Indexing. In Proceedings: 2003 Symposium on Document Image Understanding Technology, Greenbelt MD, April 2003 87-91.
- [2] J. Jeuring, Polytypic Pattern Matching, In Proceedings of the Seventh International Conference on Functional Programming Languages and Computer Architecture, ACM (1995) 238-248.
- [3] A. Singhal and F. Pereira, Document Expansion for Speech Retrieval, Proceedings SIGIR (1999) 34-41.
- [4] R. Baeza-Yates and G. Navarro, A Practical Index for Text Retrieval Allowing Errors, In CLEI, volume 1 (1997) 273-282.
- [5] S. Deerwester et al. Indexing by Latent Semantic Analysis, Journal of the American Society for Information Science, 41(6), pp. 391-407.
- [6] T. Landauer and D. Lanham. Learning Human-like Knowledge by Singular Value Decomposition: a Progress Report, Advances in Neural Information Processing Systems 10 (1998), Cambridge: MIT Press, pp. 45-51.
- [7] K. Nigam. Using unlabeled data to improve text classification, PhD. Thesis, Carnegie Mellon University, May 2001.
- [8] S. Rice, F. Jenkins, T. Nartker. The Fifth Annual Test of OCR Accuracy. Technical Report TR-96-01, Information Science Research Institute, University of Nevada, Las Vegas, NV, 1996.
- [9] D. Lewis. Reuters-21578 Text Categorization Test Collection. Distribution 1.0. README file (version 1.2). Manuscript, September 26, 1997. <http://www.daviddlewis.com/resources/testcollection/reuters21578/readme.html>.
- [10] D. Lewis, Y. Yang, T. Rose, F. Li. RCV1: A New Benchmark Collection for Text Categorization Research. Journal of Machine Learning Research, 5(2004):361-397, 2004. <http://www.jmlr.org/papers/volume5/lewis04a/lewis04a.pdf>.
- [11] H. Baird, Document Image Models and Their Uses, ICDAR, Proc. Intl. Conf. Document Anal. Recog. (1992) 62-67.
- [12] 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), (1999) 42-49.
- [13] Y. Yang. An Evaluation of Statistical Approaches to Text Categorization, Journal of

Appendix A

China wheat purchases reach 80 pct of 1996 target. China's total state purchases of wheat reached 18.53 million tonnes by the end of August, accounting for 80.26 percent of the target for 1996, the official People's Daily said on Sunday. Of the total, purchases of wheat at state-fixed prices were 13.11 million tonnes, more than 90 percent of the year's target, the newspaper said. The state also buys at market prices. The state bought 1.95 million tonnes of rapeseed, fulfilling 66.21 percent of state plans, it said. China's grain output was expected to be a record 475 million tonnes in 1996, the Xinhua news agency said on Sunday.

Figure A-1. Sample document with no errors.

China wheat purchases reach 80 pct of 1996 target. China's total state purchases of wheat reached 18.53 million tonnes by the end of August, accounting for 80.26 percent of the target for 1996, the official People's Daily said on Sunday. Of the total, purchases of wheat at state-fixed prices were 13.11 million tonnes, more than 90 percent of the year's target, the newspaper said. The state also buys at market prices. The state bought 1.95 million tonnes of rapeseed, fulfilling 66.21 percent of state plans, it said. China's grain output was expected to be a record 475 million tonnes in 1996, the Xinhua news agency said on Sunday.

Figure A-2. Sample document with 5% errors.

China wheat purchases reach 80 percent of 1996 target. China's total state purchases of wheat reached 18.53 million tonnes by the end of August, accounting for 80.26 percent of the target for 1996, the official People's Daily said on Sunday. Of the total, purchases of wheat at state-fixed prices were 13.11 million tonnes, more than 90 percent of the year's target, the newspaper said. The state also buys at market prices. The state bought 1.95 million tonnes of rapeseed, fulfilling 66.21 percent of state plans, it said. China's grain output was expected to be a record 475 million tonnes in 1996, the Xinhua news agency said on Sunday.

Figure A-3. Sample document with 15% errors.

China wheat purchases reach 80 percent of 1996 target. China's total state purchases of wheat reached 18.53 million tonnes by the end of August, accounting for 80.26 percent of the target for 1996, the official People's Daily said on Sunday. Of the total, purchases of wheat at state-fixed prices were 13.11 million tonnes, more than 90 percent of the year's target, the newspaper said. The state also buys at market prices. The state bought 1.95 million tonnes of rapeseed, fulfilling 66.21 percent of state plans, it said. China's grain output was expected to be a record 475 million tonnes in 1996, the Xinhua news agency said on Sunday.

Figure A-4. Sample document with 30% errors.

Effect of Degraded Input on Statistical Machine Translation

Faisal Farooq¹

CEDAR, SUNY at Buffalo
Amherst, NY 14228
ffarooq2@cedar.buffalo.edu

Yaser Al-Onaizan

IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
onaizan@us.ibm.com

Abstract

In this paper, we study the effects of degraded (noisy) input on the accuracy of Natural Language Processing (NLP) systems such as Machine Translation (MT). The noisy input is often the only available input as it is the output of another NLP application such as an Optical Character Recognition (OCR) system. We propose a method for reducing the noise in the input by correcting the output of an OCR system. We show that correcting OCR output reduces the degradation of the translation quality of a state-of-the-art Statistical Machine Translation (SMT) system when the corrected output is used, instead of the original output, as input to the MT system. We use a novel method for correcting the input based on a direct “phrase-based” translation system in contrast to traditional HMM source-channel models.

1 Introduction

Many NLP systems such as Information Retrieval or Speech-to-Speech Translation systems are multi-modal, where the input is often in different multimedia format such as scanned images of written text, audio, or video files. The building blocks of such systems are often other NLP applications pipelined together to form the wider NLP system. For instance a cross-lingual information retrieval system might be composed of an optical character recognition (OCR) system, a machine translation system, and a search engine. Therefore, such components must be robust against noisy input.

Component systems are often developed independently and possibly by different groups. In most cases, the internals of one component are not accessible to the developers of the next component in the pipeline. In such cases, these components (e.g. OCR) must be treated as black boxes where only their output is observed (Figure 1a). Unfortunately,

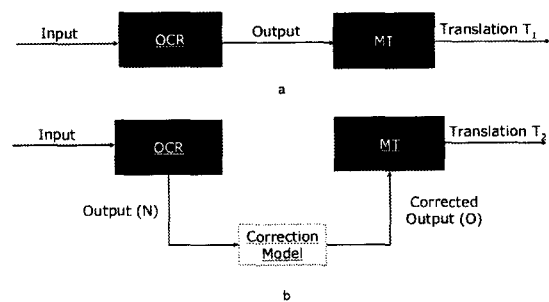


Figure 1: a) Typical NLP architecture b) NLP architecture with proposed correction model.

the output of these systems is error-prone, especially for non-Roman languages (e.g., Arabic) which have not been researched as extensively. Therefore, the next component must be robust enough to handle noisy input. The problem is compounded when the input to OCR systems, in the form of images, is also noisy.

In this paper, we describe a probabilistic model for correcting the output of OCR systems in order to minimize the degradation of translation quality of our Statistical Machine Translation as shown in Figure 1b. We discuss the manner in which such models can be trained. We evaluate the utility of our correction model by using it to correct the output of an off-the-shelf, commercially-available, state-of-the-art Arabic OCR system. In section 2 we describe the related work. In section 3 we introduce our proposed technique. We describe our training data, test data and experimental settings in sections 4.1, 4.2 and 5. We describe our models in sections 6 and 7 and the evaluation of our system performance in section 8. We finally conclude in section 9.

¹The author was a summer intern at IBM T.J. Watson Research Center, NY

Table 1: OCR Error Rates for the three datasets.

	Training Data	Development Test Set	MT'03 Eval Set
WER(%)	25.20	21.21	28.87
CER(%)	10.60	8.16	11.00

2 Related Work

There has been some work on post-processing of the OCR output. A general survey of the research in this area can be found in [1]. Jones et al. [2] describe a multi-pass OCR post-processing system which carries out individual word corrections, combined edit distance corrections and bigram probability based correction in different passes. Perez-Cortes et al. [3] use a stochastic finite state machine to test hypothesis of words. If the machine accepts the word, then no correction is made, otherwise smallest set of transitions that could not be traversed show the most similar string in the model. Pal et al. [4] describe a method for OCR error correction of Devanagiri script using morphological parsing. Some of these techniques are highly dependent on the language and use features that are specific to the language in question, which makes such techniques difficult to adapt to a different language. The method we propose here is trainable. Although we show results on Arabic OCR and the effect on Arabic-to-English translation only, the technique is adaptable to other languages where training data is available. Okan et al. [5] introduce a generative probabilistic OCR model that describes an end-to-end process in a noisy channel framework. The technique they describe is implemented as a cascade of finite state transducers. We propose a different approach to this problem. We cast this problem as a simplified translation task where the source language is the error-full OCR output and the target language is the corrected output. We also present in section 8, a more extensive evaluation of the utility of our correction technique in a more realistic NLP application (i.e., Machine Translation).

3 Proposed Technique

The erroneous output from the OCR can be thought of as an output from a noisy channel problem[6]. It can be considered as a black-box through which the signal (truth) when passed gets corrupted and emerges out as the degraded output. We can learn the corruption models (correction model in the direct model framework) and then use this knowledge to correct the output. We leverage our experience in building SMT systems by casting this correction

task as a simplified translation task. We propose a direct phrase-based translation approach that has proven effective in machine translation (e.g., [9], [7]).

As is done in a typical SMT system, given *sentence pairs* in the source (*foreign*) and the target (*english*) languages, we first align the source and target words. Word alignments can be obtained using one of several techniques such as HMM alignments [8], Maximum Entropy alignments [10], or Maximum-Posterior alignments [11]. Once a sentence pair is word-aligned, we extract phrase pairs (also referred to as block in the statistical machine translation literature) in a manner similar to [9]. These phrases are then used in combination with an n -gram language model to translate the source language into the target language.

We model the correction as a simplified translation task. The source language in our case is the OCR output (error-full) and the target language is the truth (correct). Since there is no word re-ordering as is the case of a typical MT task, the alignments can be obtained easily by minimum edit distance alignments or monotone HMM alignments. We used a simple dynamic programming minimum edit distance alignment. We approached this task in two ways. First, where the correction was achieved a character at a time, i.e., in terms of translation, the units of translation were characters instead of words as in a typical SMT system. The dynamic programming alignment between the truth and the corresponding OCR output was obtained at a character level. In the second method the same technique was applied, however, the units were words. Thus, we implemented two models - a character-based and a word-based. These models are explained further in sections 6 and 7.

4 Training and Test Data

4.1 Training Data

We use a commercially available Arabic OCR to recognize Arabic machine printed images for which we have the ground-truth. In order to create our correction training data, images of Arabic documents were needed. Since we did not have access to any publicly available large dataset of Arabic images and the corresponding text version, we automatically gener-

ated images from the arabic text of the LDC parallel corpus [14]. We used a Microsoft Word macro to open the *UTF-8* Arabic text files, format them using Bold Arabic Transparent Font at 14pt. and print them into postscript files. The postscript files were then converted into jpg images at 300 dpi. This gave us a rich source of ~ 4000 images containing 23205 segments or $\sim 1.1M$ words. These images were recognized using the Arabic OCR and the OCR output, paired with the truth, was used as the training data for our models.

4.2 Test Data

The development and blind test sets were created in the same manner described above. The development test set was created from editorials from an online Arabic newspaper (www.asharqalawsat.com). It consists of 390 segments, $\sim 11K$ words. The blind test set was the NIST MT'03 [15] evaluation set. This test set is typically used to report the quality of machine translation systems. This set consisted of 663 segments or $\sim 15K$ words. This set was used two-fold 1) as a blind test set for the correction task, 2) to test the effect of input degradation on the SMT system. This was achieved by calculating the MT quality for the original set, then the quality of MT by translating the OCR output acquired from the corresponding images and finally for the corrected output.

5 Experimental Settings

We use the evaluation metrics reported by [5] for OCR evaluation. The evaluation metrics are the word error rate (WER) and the character error rate (CER) and are standard in measuring the accuracy of OCRs. The metrics are defined as follows:

$$WER = \frac{WordEditDistance(W_{truth}, W_{Output})}{|W_{truth}|}$$

$$CER = \frac{CharEditDistance(Truth, Output)}{|Truth|}$$

where edit distance is the Levenshtein Edit Distance and is defined as the minimum number of point mutations required to change one unit(character or word) into another unit, and a point mutation is one of - substitute, insert or delete a unit. $|Truth|$ and $|W_{truth}|$ is the number of characters and words respectively in the ground-truth. The error rates for the OCR for the three sets is given in Table 1.

Our correction model is evaluated first by measuring the reduction in recognition error rates. The error reduction is measured by comparing the error rate in N and O shown in Figure 1. Since MT is word-based, and any improvement in the WER

Input String	
Arabic	وكان تم نقل التكريتي مع 17 سفيرا عراقيا
Romanized	wkAn tm nql Altkryty mE 17 sfyrA ErAqyA
Output Character Stream	
Arabic	وكان <bs> تم <bs> نقل <bs> ال تكريتي <bs> عراقي
Romanized	wkAn <bs> tm <bs> nql <bs> Altkryty <bs> mE <bs> 17 <bs> sfyrA <bs> ErAqyA

Figure 2: Conversion of input into a character stream.

would potentially affect the MT performance, in our further discussion we focus solely on the WER of the OCR or the corrected output. Secondly, we evaluate our correction model in the machine translation context by measuring the overall improvement in MT when the corrected OCR output is used instead of the original OCR output. This is achieved by comparing the translation quality (as measured by BLEU [12]) of T_1 and T_2 shown in Figure 1.

6 Character-based Model

Most OCR systems are conventionally segmentation based i.e. the text lines are segmented into words and words into their constituent characters prior to actual recognition. The OCR usually repeats the errors it makes in recognizing a certain character. In order to model the character-based nature of the OCR system, we defined the correction task as a translation of the character sequence that the OCR produces into the corresponding ground truth. Thus, the units of our input are characters instead of words in the case of a typical MT task.

6.1 Conversion to Character Stream

The first step for this task requires conversion of the input text into a stream of characters. This was achieved by separating individual characters by spaces. We used a special symbol <bs> to denote the word boundaries in order to differentiate it from the space between the characters. Figure 2 shows the same example of conversion in Arabic and romanized for illustration.

This was done for segments on both sides - the ground truth as well as the OCR output. Thus, our parallel corpus consisted of a character stream and its corresponding output from the OCR. It should be noted that the conversion on both sides into characters was performed after and not before recognition.

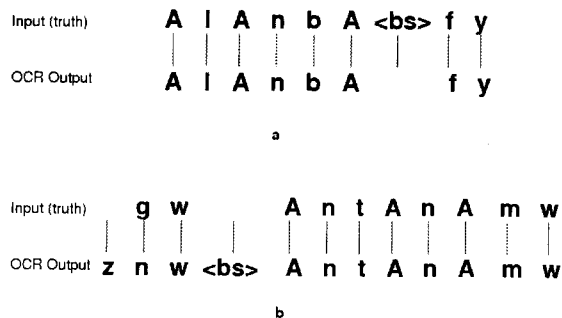


Figure 3: Examples of merge/split errors.

6.2 Training

In order to extract the phrases from the training data we align the truth and the OCR output character streams obtained above. Since there is no word re-ordering, we use a dynamic programming based minimum edit distance alignment. The edit distance was calculated using the regular Levenshtein edit distance formula with equal cost for deleting, inserting or substituting a unit. Using this, one-to-one alignments are obtained for the paired truth and output datasets. An unaligned character in the truth models a character deletion. Similarly an unaligned character in the OCR output models an insertion. Moreover, an unaligned `<bs>` in the truth models a merge error where the OCR merges two words and an unaligned `<bs>` in the OCR output models a word split. Figure 3a shows an example where two words were merged by the OCR into one and the `<bs>` is unaligned. Similarly, Figure 3b shows an example where a word was split and a character inserted (`z`). This time the `<bs>` on the OCR side is unaligned.

Using this alignment we extract all phrase pairs of lengths 1 through n ($n = 6$ in our case) characters. Figure 4 shows example phrases extracted from the training data.

6.3 Decoding

To correct the OCR output for a given test sentence, we “translate” this sentence by decoding using two weighted components - the phrases obtained above and the language model. We trained a character trigram language model by converting the gigaword corpus [13] into characters using the same procedure as above. The original corpus has 319 files totalling $\sim 4.3GB$ and $\sim 39.1M$ words. The language model is an interpolated language model. The *lambdas* were calculated from held out data, which is the last 10% of the gigaword corpus.

Source (OCR)	Target (Truth)	Count
ن ي (n y)	ن ي (n y)	17437
ن ي (n y)	غ ي (g y)	22
ن ي ب ي (n y b w)	ن ي ب ي (n y b w)	2
ن ي ب ي (n y b w)	ن ي ق ي (n y q w)	2
ن ي ب ي س (n y b w s)	ن ي ق ي س (n y q w s)	35

Figure 4: Phrases extracted from training data after conversion to character stream.

The decoder is a beam search decoder. For the correction problem, the decoding is carried out in a monotone fashion such that characters at the beginning of the sentence must be decoded first. Decoding hypotheses are extended in cardinality-synchronous fashion such that at any given time step n , only hypotheses that cover exactly n characters (i.e., of cardinality n) are extended. For example, at time step 0 only hypothesis of cardinality 0 (in the special init hypothesis) are extended. The decoding is continued until all characters in the sentence are covered. The final decoding is the hypothesis of cardinality l with the least cost, where l is the length of the source text. More formally, the final decoding ϵ for the source f is the one that satisfies the following equation:

$$\epsilon = \underset{\epsilon}{\operatorname{argmax}} [-w_{ph} \times \log_{10} P(\epsilon|f) - w_{lm} \times \log_{10} P(\epsilon)]$$

where $P(\epsilon)$ is the trigram character language model probability and $P(\epsilon|f)$ is the phrase-based direct model. The weights w_{ph} and w_{lm} of these components were trained using the first 50 segments of the development test set. the weight are trained using maximum BLEU training such that weights are adjusted to maximize the BLEU score. The weights that resulted in the maximum BLEU score for these 50 segments were selected for correcting the rest of the test set and any subsequent test sets using the monotone decoding as described earlier. Since, there were only two weights to be trained, we also adopted a grid-search technique. The translation accuracy(measured by WER) was calculated by varying the phrase component weight from 1.0 to 0.5 in steps of 0.1 and correspondingly the language model weight was varied from 0.0 to 0.5.

Table 2: Reduction in WER of development test set using word-based model before and after normalization.

	OCR Output	Corrected Output	Improvement (abs.)	Improvement (rel.)
WER % (Before Norm.)	21.21	20.08	1.13	5.32
WER % (After Norm.)	17.7	13.4	4.3	24.29

6.4 Results

The character-based model resulted in a 1% absolute (12% relative) reduction in the CER of the development test set. There was no improvement in the WER, which remained at 21.21%. This could be attributed to a number of reasons. In addition to the character trigram language model being weak, the phrase features were also not helpful. It was noted that the OCR had a post-processing step where the recognized output was converted to the closest possible dictionary word. This led to the fact that besides correctly recognizing the character, the rest of the confusion matrix was rather smooth than sharp as expected making the phrase features less reliable. Due to this post-processing, the actual error that the OCR would make on a character was hidden. By the creation of a dictionary word, the OCR output character could be replaced by another character and thus would not make it possible to model the error the OCR would make often.

7 Word-based Model

As noted in section 6.4, the character-based model proved to be weak to correct OCR errors. To overcome this, a word-based model is pursued, which proved to be useful. The word trigram language model would be much more beneficial since it provides more context.

7.1 Training

The training data was again aligned using the minimum edit distance, however, only this time based on words rather than characters. After aligning the truth and the OCR output, we extracted word phrases just as we extracted character phrases as described earlier in section 6.2. The drawback with this, however, would be that in order to get features for all possible words, we need much more data. In order to overcome this, we adopted the following procedure:

1. Extract phrasepairs of length=1 pairs $P = \bigcup p_w$, where p_w is a phrase pair containing source word w and all target word(s) from the training data.

2. Get the vocabulary V_t of the test set.

3. $\forall w_i \in V_t$, if $\exists p_w \in P$ such that $count(p_w) < T$ then $P = P \cup \{w_i, neighbour(w_i)\}$, i.e. add w_i and $neighbour(w_i)$ to P .

Given a very large language model vocabulary V_{lm} , the function $neighbour(w_i)$ yields all words $w'_i \in V_{lm}$ such that the minimum edit distance of w'_i from w_i is less than a threshold T' . For example, if the OCR output word was *AntAnAmw* (antanamo), $neighbour(AntAnAmw)$ would yield $\{AntAnAmw, gwAntAnAmw, ntAnyAhu, \dots\}$ (antanamo, guantanamo, netanyahu, ...). The thresholds T and T' were chosen empirically.

7.2 Decoding

Translation was carried out in the same manner as in case of a character-based model, however, the units this time were actual words. For this we trained a word trigram language model from the gigaword corpus[13] in a similar fashion as in section 6.3. However, this time there was no conversion into character streams. The weights for the phrase and the language model components were obtained by the grid search technique described in section 6.3.

7.3 Results

The word model performed better than the character model in general (1.13% for the dev. test set and 13.71% for the blind test set absolute improvement in WER as compared to no improvement by the character model). Since Arabic is highly inflectional, many of the errors were prefix/suffix errors. Our trigram language model had difficulty predicting the right set of prefix/suffix combination to choose for Arabic words. Other errors were less severe. For example, confusing various forms of *alef* (i.e., *alef* without *hamza* or with *hamza* above or below it). Since various forms of *alef* are mapped to one canonical form by the MT engine (in a process we call *normalization*), these errors were ignored by normalizing the OCR output as well as the corrected output that our model generated. The WER of the OCR after normalization was 17.7%. However, after normalizing the corrected output, the WER dropped

Table 3: Reduction in WER of MT03 test set using word-based model before and after normalization.

	OCR Output	Corrected Output	Improvement (abs.)	Improvement (rel.)
WER % (Before Norm.)	28.87	15.28	13.59	47.07
WER % (After Norm.)	13.71	11.54	2.17	15.83

from 20.08% to 13.4%. Thus, we were able to get a 4.3% absolute reduction in the WER after correction compared to 1.13% absolute reduction before normalization as shown in Table 2.

A similar improvement is also observed for our blind test set. The WER before normalization was reduced from 28.87% to 15.28%, and from 13.71% to 11.54% after normalization as shown in Table 3.

8 Evaluation

In order to evaluate the effect of errors introduced by the OCR in an Arabic to English Machine Translation system, we calculated the quality of MT before correction and after correction of the OCR output for the MT03 dataset. The translation is performed on the normalized text and the BLEU scores are calculated with the NIST provided reference[15]. Normalization is carried out as described in section 7.3. Table 4 shows the BLEU score for translation of the MT03 test set before and after correction. The BLEU score for the actual MT03 test set(truth) is also reported for comparison. As shown we achieve a statistically significant improvement in the BLEU scores.

9 Conclusion

We described a novel system for correcting the erroneous output of any OCR system. Our methods are generic and can be extended to any language. We achieved considerable reduction in word error rate. In addition, we demonstrated the utility of our system in a NLP application. OCR errors reduce the machine translation quality. However, this degradation can be minimized when our correction model is used to post-process the output of the OCR system. We were able to get a statistically significant increase in BLEU scores when our correction model is used (0.401 vs. 0.378).

However, the BLEU score obtained when translating the original text (truth) is still better than that of the corrected OCR output (0.465 vs. 0.401). A further reduction of the OCR word error rate might be necessary to reduce that gap. We believe that acquiring more training data for the word model will further reduce the error rates. The training data can

be easily generated in the manner described in this paper. Such data can be easily obtained for any language or domain given that enough text is available electronically for that language/domain.

We also believe that our character-based model can be improved if used in conjunction with a word-based language model, instead of the character-based language model that was used.

References

- [1] K. Kukich, Techniques for automatically correcting words in text, *ACM Computing Surveys* (1992) 24(4), 377-439.
- [2] M.A. Jones, G.A. Story and B.W. Ballard, Integrating multiple knowledge sources in a Bayesian OCR post-processor, *International Conference on Document Analysis and Recognition* (1991) 925-933, St. Malo, France.
- [3] J.C. Perez-Cortes, J.C. Amerngual, J. Arlandis and R. Llobet, Stochastic error-correcting parsing for OCR post-processing, *International Conference on Pattern Recognition* (2000) 4405-4408, Barcelona, Spain.
- [4] 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.
- [5] O. Kolak, W. Byrne and P. Resnik, A Generative Probabilistic OCR Model for NLP Applications, *Proceedings of HLT-NAACL* (2003) 55-62, Edmonton, Canada.
- [6] L.R. Bahi, F. Jelinek, R.L. Mercer, A Maximum Likelihood Approach to Continuous Speech Recognition, *IEEE Transactions on PAMI* (1983) 5(2), 179-190.
- [7] P. Koehn, F.J. Och and D. Marcu, Statistical Phrase-Based Translation, *Proceedings of the Human Language Technology Conference (HLT-NAACL)* (2003), Edmonton, Canada.

Table 4: BLEU scores for the MT03 test set(± 0.0164 for statistical significance).

	Ground Truth	OCR Output	Corrected Output
BLEU Score	0.4651	0.3776	0.4013

- [8] S. Vogel, H. Ney, and C. Tillmann, HMM Based Word Alignment in Statistical Translation, *Proceedings of the 16th International Conference on Computational Linguistics (COLING'96)* (1996), 836–841, Copenhagen, Denmark.
- [9] C. Tillmann and H. Ney, Word Re-ordering and a DP Beam Search Algorithm for Statistical Machine Translation, *Computational Linguistics* (2003) 29(1), 97–133
- [10] A. Ittycheriah and S. Roukos, A Maximum Entropy Word Aligner for Arabic-English Machine Translation, *HLT-NAACL* (2005), Vancouver, Canada.
- [11] B. Zhao, N. Ge and K. Papineni, Inner-Outer Bracket Models for Word Alignments using Hidden Blocks, *HLT-NAACL* (2005), Vancouver, Canada.
- [12] K. Papineni, S. Roukos, T. Ward and W. Zhu, BLEU: a Method for Automatic Evaluation of Machine Translation, *Proceedings of the 40th Annual Meeting of Association of Computational Linguists (ACL)* (2002), 311–318, Philadelphia, USA.
- [13] Arabic Gigaword Corpus, *Linguistic Data Consortium (LDC)*, <http://www ldc upenn edu> (2003) Catalog LDC2004T17, Philadelphia, USA
- [14] Arabic News Translation Text, <http://www ldc upenn edu> (2004) Catalog LDC2003T12, Philadelphia, USA
- [15] MT'03 Arabic to English MT Evaluation Set, <http://www ldc upenn edu/> (2003), Philadelphia, USA

Mobile Interactive Support System for Time-Critical Document Exploitation

George Nagy

Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute
Troy, NY 12180
nagy@ecse.rpi.edu

Daniel Lopresti

Department of Computer Science
Lehigh University
Bethlehem, PA 18015
lopresti@cse.lehigh.edu

Abstract

Computer Assisted Visual Interactive Recognition, or CAVIAR, has proven to be an effective methodology for integrating human and machine expertise in addressing challenging pattern recognition tasks. In this paper, we examine the application of the CAVIAR paradigm to problems arising in real-time document analysis in the field. We identify hurdles we expect to encounter, propose potential solutions, and describe an evaluation framework to help determine whether this is a fruitful line of research.

1 Introduction

Research aimed at fully automating the processing of document images has received a tremendous amount of attention over the past 40 years. As a quick perusal of any of the dozens of surveys to date will reveal, however, progress in automatic recognition and interpretation has been slower than predicted. We expect that further improvement in accuracy in domains such as cursive handwriting and degraded documents will be even more protracted because the challenges that remain are much harder. As in speech recognition, bridging the gap between machine and human knowledge to allow the former to draw even with the latter appears problematic. The context brought by humans to any classification task is much greater than what can be obtained from even the largest collections of training samples available to our community. Endowing fully automated systems with broad knowledge remains a far-off goal.

There exists, however, a significant body of applications where it is not necessary to fully automate the task of document analysis. Rather, the focus is on a relatively small number of high-value documents (perhaps just one) – say from a cache discovered in the field as part of an ongoing criminal justice, military, or intelligence operation – where the computer plays the role of an assistant to help

the user acquire information that would otherwise remain inaccessible. While such documents could be collected and returned to a central repository for scanning and batch processing in the traditional manner, there is often a significant advantage in being able to exploit the information in real-time, immediately and in situ.

Recently, Nagy and Zou, et al. have begun exploring a concept known as CAVIAR: Computer Assisted Visual Interactive Recognition [3, 6, 24, 23]. Over the last few years this work has led to the development of a successful interactive system for recognizing faces and flowers, both problems of a level of difficulty (i.e., automated current accuracy) comparable in certain ways to document recognition in the field. Experiments on sizable databases of faces and flowers indicate that interactive recognition is more than twice as fast as the unaided human, and yields an error rate ten times lower than state-of-the-art automated classifiers. The benefit margin of interactive recognition increases with improved automated classification. Parsimonious human interaction throughout the interpretation process is much better than operator intervention only at the beginning and the end, e.g., framing the objects to be recognized or dealing with rejects. Furthermore, this interactive architecture has been shown to scale up: it can start with only a single sample of each class, and it improves as recognized samples are added to the reference database.

The notions embodied in CAVIAR differ in fundamental ways from past efforts at mobile and/or interactive recognition. Whether such an approach can be equally effective in the domain of documents as it is for flowers and faces is unproven. In this paper, we discuss the application of this paradigm to document analysis in the field, identify the hurdles we expect to encounter, propose potential solutions, and describe an evaluation framework to help deter-

mine whether this is a fruitful line of research.

2 Related Work

The questions we plan to examine arise from adapting CAVIAR to tasks from document analysis. There are, however, other projects that share similar goals and assumptions; below we briefly cite a few of these. The Army Research Laboratory's Forward Area Language Converter (FALCon) system provides mobile optical character recognition (OCR) and translation capabilities [10, 21], but, so far as we know, employs a traditional user interface. A growing amount of research is being conducted on camera-based document acquisition (e.g., [7, 11]). This work employs the camera as a new form of capture device, but, as with FALCon, treats the later processing stages as though they will be fully automated

Camera-based systems for locating and recognizing text in traffic signs and providing translation services for non-native visitors in foreign lands are perhaps more similar to what we have in mind [8, 22]. Still, we have yet to encounter such a system with an interaction paradigm as integral and as sophisticated as CAVIAR's. Reading services for the vision-impaired are likewise focused on page-at-a-time processing, but employ an auditory user interface as opposed to a visual one for obvious reasons [9, 19]. A somewhat similar notion is recent work on developing tools to support forensic document analysis [20]. We note, however, that such systems are designed to solve a different, much more specific problem, are intended for off-line use by domain experts (as opposed to occasional users whose primary jobs lie elsewhere), and have no need for mobility.

3 Rationale for a Human-Machine Collaborative Approach to Document Analysis

A divide-and-conquer strategy for visual recognition should partition difficult domains into components that are relatively easier for both human and machine. There are pronounced differences between human and machine cognitive abilities. Humans excel in gestalt tasks, like object-background separation. We apply to recognition a rich set of contextual constraints and superior noise-filtering abilities. Computer vision systems, on the other hand, still have difficulty in recognizing "obvious" differences and generalizing from limited training sets. We can also easily read degraded text (e.g., CAPTCHA's [2]) on which the best optical character recognition systems produce only gibberish.

Computers, however, can perform many tasks faster and more accurately. Computers can store thousands of images and the associations between them, and never forget a name or a label. They can compute geometrical properties like higher-order moments whereas a human is challenged to determine even the centroid of a complex figure. Spatial frequency and other kernel transforms can be easily computed to differentiate similar textures. Computers can count thousands of connected components and sort them according to various criteria (size, aspect ratio, convexity). They can quickly measure lengths and areas. They can flawlessly evaluate multivariate conditional probabilities, decision functions, logic rules, and grammars. On the other hand, the study of psychophysics revealed that humans have limited memory and poor absolute judgment [16]. A detailed comparison of these differences appears in Table 1.

4 Technological Issues

Here we briefly summarize some of the technological issues that must be addressed in the implementation of a CAVIAR-like system to support document analysis and exploitation:

1. The rapid development of high-quality, low-cost digital cameras suitable for full-page imaging in color or a broad range of grays. Smaller versions of these cameras are now available as plug-ins for Personal Digital Assistants (PDA's) and within a year sufficient resolution should be available for camera phones. These devices make document-acquisition in the field simpler than with conventional page-width scanners.
2. Displays are approaching the resolution-limit of the human eye (discounting head movement). In addition to the small displays built into PDA's, cell phones, and helmets, larger flexible displays that can be incorporated into a user's clothing are on the verge of becoming available.
3. Interaction with a direct-action device like a stylus, and especially a thumb, is faster than with a mouse. Touch-sensitive screens are just now becoming available for cell phones.
4. The storage capacity of mobile devices is already sufficient for most DIA and OCR tasks.
5. Nagy, et al. have implemented interactive recognition on both stand-alone and wireless networked mobile platforms. Both are adequate for PDA's, but neither is yet acceptable for cell-phone based systems because these do not have

Table 1: Comparison of relative strengths of human vs. machine in visual pattern recognition.

Human	Machine
<ul style="list-style-type: none"> • dichotomies • figure-ground separation • part-whole relationships • salience • extrapolation from limited training samples • broad context • gauging relative size and intensity • detection of significant differences between objects • colored noise, texture • non-linear feature dependence • global optima in low dimensions 	<ul style="list-style-type: none"> • multi category classification • nonlinear, high-dimensional classification boundaries • store and recall many labeled reference patterns • accurate estimation of statistical parameters • application of Markovian properties • estimation of decision functions from training samples • evaluation of complex sets of rules • precise measurement of individual features • enumeration • computation of geometric moments • orthogonal spatial transforms (e.g., wavelets) • connected component analysis • sorting and searching • rank-ordering items according to a criterion • additive white noise • salt & pepper noise • determination of local extrema in high-D spaces

enough computational resources for stand-alone operation, and their bandwidth is too low for acceptable interactive response time on image computations. However, according to technology forecasts, both of these shortcomings will disappear in a year or two.

6. The need for network connectivity depends greatly on the targeted application domain: civilian, military, or covert. While it is clearly desirable for such a system to be operable completely autonomously, there may be substantial value in networking document acquisition and exploitation activities.

5 Data Entry on Hand-Held Devices

It is clear that an important constituent of mobile interactive document analysis is manual text entry. The alternatives seem to be restricted to virtual keyboard on a touch sensitive screen activated by a handheld stylus, finger-operated keyboards incorporated in the operators clothing (on arm or thigh), and automatic speech recognition. We believe that the stylus is the most appropriate solution, because in addition to text entry it can also mediate the graphical communication essential in some aspects of document image analysis.

The virtual keyboard appeared in the seventies to accelerate the digitization of maps and line-drawings by avoiding having the operator shift constantly between pointing device and keyboard. It consisted of a picture of a keyboard on a piece of paper that could be shifted to the area of the drawing being vectorized. The current virtual keyboard usually appears in a fixed partition of the touch-sensitive screen of a handheld device. The text being entered appears on another partition. Edwards surveyed input interface issues in mobile devices in 1997 [4]. Data input is usually a local operation, so it makes little difference whether the device is networked or not.

The key consideration for stylus data entry are speed, (perceived) operator comfort, and ramp-up time. The first two factors are influenced by the amount of space allocated to the keyboard, to the recognized or entered text, and to control functions. The third factor depends heavily on the keyboard layout. The QWERTY layout, developed to prevent binding of type bars in mechanical typewriters, is suboptimal even for typing, and even more so for one-handed stylus entry.

There has been much work on evaluating alternative keyboards and word-completion algorithms (for a recent overview, see [1]). An upper bound on the speed of individual character entry is imposed

by Fitts' Law, which is a nonlinear relationship between pointing time and the distance and size of the target. The relevant distance is that between the screen areas ("keys") corresponding to consecutive letters. The letter transition frequency is given by a language model. As an example of the compromises necessary in keyboard design, it is possible to reduce the average distance by having several space keys, but this decreases the size of the keys. Several researchers have optimized keyboard designs according to various language models [14, 13, 15, 12]. The computed speeds hover about 40 words per minute, but actual text entry is much slower.

The speed increase obtainable by word completion also depends on the language model. Ancona [1] demonstrates a keyboard with separate keys for the ten most common words (with a cumulative word frequency of 28%). After each tap on the screen, the ten most likely words appear in the selection area of the screen. If the correct word is included, it can be selected with one additional tap. If not, another letter is tapped, which brings up ten new words. With a vocabulary of 13,000 words, the expected number of taps per word is claimed to be 3.3. Some of the notions here are clearly similar to those incorporated in CAVIAR.

The performance of word-completion systems depends on how well the stored lexicon is matched to the user input. Multiple lexicons – for different languages and applications – can be either stored on board, or downloaded via a wireless connection.

Another important source of ideas is the technology developed for vectorizing maps and engineering drawings [17, 18]. Manual vectorization was first conducted from hardcopy with a digitizing table or tablet. The operator traced the lines with cross-hairs under a magnifying glass with a MARK button. After the advent of large-size roller-feed scanners and bitmapped displays, all service bureaus and in-house operations converted to on-screen vectorization from scanned copy. One advantage was that already vectorized lines could be displayed with a different color, and departures between manually entered lines and original were clearly visible. The operator could zoom in on dense portions of the drawing. However, the aspects of interest here are the algorithms developed for semi-automated data entry.

For colored maps, different color layers were first separated according to RGB values. Vectorizing algorithms were manually initialized to a line segment or curve, and then could automatically follow that line at least to the next intersection point. The systems would also attempt to automatically recognize map and drawing symbols (for schools or re-

sistors). If it failed, the operator would override it. The character recognition software recognized cleanly lettered labels (elevations, part numbers, resistor values), but left labels confused by overlaid line art or poor lettering to the operator.

If most of the labels cannot be recognized by OCR because of poor document quality or unusual character shapes, it is still possible to rapidly mark their location and orientation, rotate them to horizontal, and move them to a single area of the screen. This accelerates manual label entry. A single E-sized drawing may contain 3,000 alphanumeric symbols, which is more than a densely printed page of text.

Most such data-entry systems are part of larger GIS or CAD software, and are typically designed for standard workstations. All graphical operator interaction is therefore mediated by the mouse. As demonstrated by Engelbart and colleagues at SRI long ago, direct-action devices, like a touch-sensitive stylus, allow faster and more accurate interaction [5]. Indeed, we found that to be the case in comparing desk-top CAVIAR systems with handheld CAVIAR.

Like CAVIAR, these interactive systems exhibit clear speed advantages over completely manual data entry, and are robust enough (unlike automated systems) for operational application. Although some of these systems are laboriously trainable, one key difference compared to CAVIAR is that no commercial system that we are aware of incorporates adaptive algorithms that take advantage of routine operator input. Unlike CAVIAR, they also fail to provide visible models for the entry of complex 2-D patterns.

6 Test Data Collection

Any comparison of the proposed mobile interactive document exploitation system with existing and forthcoming automated and manual data entry systems requires a test database. It is, of course, desirable that the test database reflect the characteristics of the documents that are to be processed. For the purposes of this discussion, we assume that the domain of interest is handwritten Arabic documents; however, it should be clear how the the same issues and potential solutions generalize to other genres and languages.

We believe that existing handwritten Arabic databases cannot fulfill the objectives we have in mind for CAVIAR research because they are too small, they were designed specifically for testing a particular class of recognition algorithms, and they do not reflect the characteristics of the target data. For the sake of concreteness, we make the following assumptions:

1. The proposed system is to be applied to transcription and further processing of documents similar to those in a growing collection of handwritten Arabic documents, referred to here as the *Target Database*. Although we propose a mobile, personal device operated by a non-specialist, the test database will also be used for evaluating batch-mode DIA and OCR systems.
2. The amount of data and corresponding meta-data, including ground truth (GT), will be comparable to an existing database used for evaluating typeset Arabic documents, i.e., 400,000 words. We take this as a fixed point for now. Handwritten documents have fewer words than printed text: say 100 words/document on average. The documents are not restricted to a single page, but each document is created by a single author.
3. Handwritten documents have statistically different textual content than typeset documents; therefore we cannot mirror the existing database. Handwritten material is less likely to form a grammatical narrative: it may have unusual abbreviations, short lists of phrases, unlabeled strings of digits, underlines, corrections, cross-outs and erasures, and unstylized layouts.
4. Character formation in free writing in any script is markedly different from copying or taking dictation from tape or a person. The conditions under which the document is composed, including the degree of stress or haste, will perceptibly alter even the same person's writing. We believe that this must be taken into consideration to avoid disappointment when the system is deployed.
5. The most important components of systematic variability are (a) country of education, (b) level of education (years of schooling), and (c) age of the writer.
6. Digitization and character recognition are affected by the medium: pen or pencil, n^{th} generation copy, fax, paper quality, and physical conditions (writing on a desk is different from writing in a hand-held notebook).
7. The chosen approach for (semi-)automated transcription should depend on the distribution of the amount of writing among individuals. If there are many long passages or multiple documents by the same writer, the system ought

to take advantage of this. Language context and writer/style adaptation are powerful aids to recognition.

8. A service bureau with Arabic software can perform scanning and ground-truthing more efficiently than students. Such a service bureau could be located in a country with a significant Arab-speaking population (US, Canada, UK), or in a friendly Arab country (Jordan, Egypt) with an advanced computer infrastructure.

In order to plan data collection, it would be desirable to have the following information. The relevant document statistics could either be estimated (guessed) by a designated expert, or obtained from the agency responsible for processing the target documents.

Ideally, a sizeable random sample of the original documents (or very good copies) with translations from the Target Database, with accompanying meta-data, would be made available. Most of the necessary information could be derived from such a sample. If, as is likely, such a sample cannot be released, then specification of the following distributions (in the form of histograms) would be useful:

- Number of *words per document*.
- Number of *documents per writer* and of *words per writer*.
- *Educational profile* of the writer.
- *Country of schooling* of the writer.
- *Vocabulary*: a lexicon indexed by frequency and, if possible, by writer.
- *Media*: pen, pencil, copy, fax, photo, etc.

In order to gauge feasibility, in Table 2 we provide a rough estimate (perhaps accurate to $\pm 50\%$) of the cost of deriving the required statistics from a hardcopy database of documents. These costs do not include the significant cost of collecting the documents, nor planning/management costs.

The handwriting data could consist of either: existing originals (or high-quality copies thereof), or handwritten documents generated for this purpose by about 1,000 subjects.

Possible sources of existing data include schools (assignments, term papers, course notes, administrative records), businesses (orders, reports, correspondence), and government agencies (dead files of forms and free-form correspondence). Privacy concerns may preclude certain of these approaches, however.

Table 2: Estimated costs of a hardcopy document ground-truthing activity.

Category	Cost
Scanning (200 or 300 dpi, 8-bit grayscale), for 4,000 mostly single-page documents, with quality control, at \$1.00 per document	\$4,000
Transcription, with verification and correction, at \$2.00 per document	\$8,000
System acquisition or customization for detailed (word-box level) zoning	\$10,000
Interactive box location, verification and correction	\$8,000
Database creation (accession numbers, bitmaps, boxes, GT)	\$12,000
Estimated total	\$42,000

If new data must be generated, a possible scenario for distributed data collection is:

1. Prepare from five to eight printed single or multi-page protocols on various topics, e.g., economics, politics, technology, military, religion. The topics should correspond to their proportions in the target database. The protocols should include questions that require a narrative answer, including requests for summarization, organization, decision, rebuttal, comment or memory-aids in the subjects' own words. It is essential to avoid having the subjects copy fixed material, and to give rise to varied vocabulary, syntax, spelling, and document length.
2. Recruit ~1000 Arab-literate subjects according to specified demographics, possibly in different countries, and request each to respond to one or more forms. In U.S. university settings, this would require permission for the cognizant institutional review board on human factors experimentation, and obtaining signed consent forms.
3. Provide each writer a writing surface, paper and writing instrument according to the randomized experimental design.
4. Collect the forms, possibly with the demographic data attached to each sheet. Scan the image data and organize the metadata in a widely readable format (ASCII perhaps). The size of the database after digitization at 200 dpi

(8-bit depth) should be about 16 GB. Because of the prevalence of white space, after compression it may fit on a single CD.

For DIA and OCR research and development, this method would far superior to the customary collection of copies from prescribed forms. The latter, however, would be less expensive because the handwritten data need not be transcribed but only proof-read to catch copying errors.

Part of the resulting database should be sequestered for independent testing. The remainder could be released to the research and development community. Used in this manner, the proposed database would have a relatively long shelf-life and provide performance measures far more representative of operational systems than the customary greenhouse test data.

7 Examples of Interactive Document Image Analysis

We mention some DIA tasks where automated algorithms work accurately only on exceptionally clean documents, but where a little interaction can quickly produce acceptable results.

Most OCR algorithms, especially for handwriting, are designed for binarized images, because scripts generally avoid discrimination based only on shades of gray or color. Instead of using thresholding hardware built into the scanner, today documents are usually digitized to 8-bit gray scale or RGB, and subsequently converted to binary images. Global binarization algorithms work only if the foreground and background reflectance are uniform throughout the document, which may not be the case if, for example, part of a folded documents suffers prolonged exposure to sunlight, or if there are dark areas around the edges of a photocopy. Local binarization algorithms measure the distribution of reflectance in a window translated through the page, and set the local threshold between foreground and background reflectance peaks according to the estimated dispersions of the components of the mixture distribution. The window size and reflectance distribution estimates invariably depend on explicit or implicit assumptions about the relative density and configuration (strokes) of the foreground (ink) and background. These assumptions generally hold only for a narrow class of documents. Fortunately, the binarization algorithms are simple, therefore an operator can easy set the appropriate window size and foreground density either for the whole document, or for selected areas. This still allows local algorithmic thresholding, and therefore requires much less

interaction than setting the threshold manually everywhere, and is much more robust than fully automated local thresholding.

Line finding is another instance where interaction may be effective. The first step is usually estimating global document skew, i.e., the angle of the written lines with respect to the paper or digitizer axes. While very accurate skew estimation and correction algorithms have been developed for printed matter, they do not work well on handwriting because the orientation of individual lines varies, the margins are not straight, there may be only a few words on a page, and there may be several columns of words or phrases at different angles. Humans can, however, judge skew remarkably well, and convey this information to the computer by a few well chosen stylus taps or by rotating a superimposed grid. After the computer-proposed skew correction and line finding is corrected, the occasional merged pair of lines – due to overlapping ascenders and descenders – can be likewise rapidly separated.

Word segmentation is relatively easy for printed text, except for extremely tightly-set, micro-justified print. In handwriting, however, large spaces often appear within words and, towards the end of a line, words are often squeezed together. In Arabic and other scripts, some inter-letter spaces are mandatory. Underlines that link word sequences can further complicate the task. Again, humans can usually spot missed word boundaries even in unfamiliar languages and scripts. If the writing lines are already properly segmented, then a simple interface can be designed to correct linked and broken words.

At the character recognition level, there are also several opportunities for effective interaction. First, humans can often tell where perfect accuracy is important, as in telephone numbers, email addresses, and proper nouns. If the automated algorithms fails on important words, phrases or numbers, they can be either entered manually using the virtual keyboard, or selected by a stylus tap from the top recognition candidates.

The human can provide global assistance to the character recognition system. The operator may be able to recognize the language or script of a document even from a few words, perhaps by using common sense or contextual information related to the source of the document (e.g., indicate that the document probably contains a mixture of Korean and German.) He or she can indicate the average slant, and in Western scripts, the prevalent case (e.g., if a writer uses only capital letters). The operator may also decide which of the available lexicons would provide the best language model. (The lexicons will

be automatically updated with entries from the processed documents that have been deemed correct.)

Most importantly, entering only part of a document may provide enough training data – to a recognition system designed with this in mind – for fine-tuning the classification algorithms. The underlying assumption is that if the remainder of the document (and perhaps also additional documents) is from the same source, the adjusted parameters will yield more accurate recognition. If that is not the case, the operator can easily separate the portions of the document written by different individuals (assuming that the second writer was not attempting to mimic the first). Of course, human handwritten data entry in an unfamiliar language and script is also problematic. However, it requires far less training than learning to speak, write and understand a language. Off-shore data-entry is sometimes carried out by operators who do not know English. Anecdotal evidence suggests that on printed matter at least, non-speakers are more accurate, because they must look at each letter.

8 Evaluation Plans and Issues

The requirements for evaluating a CAVIAR-like approach to document analysis differ from traditional, fully-automated techniques. In the latter case, the focus is entirely on classification accuracy, whereas in the former, the system is comprised of a human user and a machine working in collaboration: both time and accuracy are important measures. It is unlikely the system will be more accurate than a human working alone, it must be faster. Similarly, it is unlikely the system will be faster than a fully-automated algorithm, it must be more accurate. Note also that such work naturally demands the use of human test subjects, unlike most pattern recognition research.

One of the most intriguing aspects of the earlier CAVIAR work was its demonstration of an inherent potential to incorporate adaptation, yielding a system that improves with use. This also provides the opportunity to study human learning: the operator and the system learn together, a little like a blind person and her Seeing Eye dog. As was done in the case of CAVIAR for flowers, we plan to study this effect.

It is also evident that one of the most plausible targets for a CAVIAR-like system involves documents written in a language that the user is unable to read, perhaps using a script that he/she does not recognize. A typical end-to-end scenario, then, might include the following processing stages:

1. Categorization: source, age, type of document, size, quality, language, script.
2. Image-level preprocessing including various levels of segmentation.
3. Word recognition.
4. Translation.
5. Information importance assessment/transformation into actionable knowledge.

Our initial experiments will be conducted on workstations with simulated mobile-device window sizes.

Among other expected benefits, we believe that the development cycle for interactive recognition systems may be faster than for wholly automated systems. One reason to suspect this will be true is that it is no longer necessary to take care of rare contingencies: they can be dealt with by the human operator, who can always override every automated option. This will allow deployment of a whole family of support systems, specialized to different levels of operator familiarity with the target language(s) and script(s).

We intend to develop the interfaces necessary to accomplish the above interactive tasks in parallel with data collection. While this would have been a monumental endeavor only a few years ago, today excellent tools are available for the purpose. Perhaps the most difficult part of the project is devising and incorporating a customized logging system for timing and recording every interaction and its effect. This must be done in sufficient detail to allow replaying the whole interactive document information acquisition sequence in order to find out where and what goes wrong.

The individual files log must be compiled and aggregated for statistical analysis. This will also form the basis for evaluating the interactive system and for comparing it with entirely automated and entirely manual data entry.

We emphasize that we do not address the extraction of data from a large backlog of accumulated documents. Unlike much of the academic research on document analysis, our interest here is not in archival documents, but in those whose value decreases exponentially with time. We propose a system that may prevent future backlogs of nearly worthless documents by on-site, just-in-time information extraction.

References

- [1] M. Ancona, S. Locati, M. Mancini, A. Romagnoli, and G. Quercini. Comfortable textual data entry for PocketPC: the WTX system. In *Advances in Graphonomics, Proceedings of IGS 2005*, Salerno, Italy, June 2005.
- [2] H. S. Baird and D. P. Lopresti, editors. *Human Interactive Proofs: Second International Workshop*, volume 3517 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, Germany, 2005.
- [3] S-H. Cha, A. Evans, A. Gattani, G. Nagy, J. Sikorski, C. Tappert, P. Thomas, and J. Zou. Computer Assisted Visual Interactive Recognition (CAVIAR) technology. In *Proceedings of the IEEE Electro/Information Technology Conference*, May 2005.
http://www.ecse.rpi.edu/homepages/nagy/PDF_files/EIT2005-Evans_et_al.pdf.
- [4] J. Edwards. New interfaces: Making computers more accessible. *IEEE Computer*, pages 12–14, December 1997.
- [5] W. K. English, D. C. Engelbart, and M. L. Berman. Display-selection techniques for text manipulation. *IEEE Transactions on Human Factors in Electronics*, HFE-8(1):5–15, March 1967.
- [6] A. Evans, J. Sikorski, P. Thomas, J. Zou, G. Nagy, S.-H. Cha, and C. C. Tappert. Interactive visual system. Technical Report CSIS No. 196, Pace University, 2003.
<http://csis.pace.edu/~ctappert/ivs/ivs-nagyetal.pdf>.
- [7] F. Fisher. Digital camera for document acquisition. In *Symposium on Document Image Understanding Technology*, Columbia, MD, 2001.
<http://www.dtic.mil/matris/sbir/sbir022/a038.pdf>.
- [8] H. Fujisawa, H. Sako, Y. Okada, , and S. Lee. Information capturing camera and developmental issues. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition (ICDAR '99)*, pages 205–208, Bangalore, India, September 1999.
- [9] T. Hedgpeth, M. Rush, J. Black, and S. Panchanathan. The iCare project reader. In *Sixth International ACM SIGACCESS Conference on Computers and Accessibility*, October 2004.
<http://cubic.asu.edu/grants/downloads/RDE/iCareReader.doc>.

- [10] M. Holland and C. Schlesiger. High-mobility machine translation for a battlefield environment. In *Proceedings of NATO/RTO Systems Concepts and Integration Symposium*, volume 15, pages 1–3, Monterey, CA, May 1998.
- [11] C. Jacobs and P. Simard. Low resolution camera based OCR. *International Journal on Document Analysis and Recognition*. To appear.
- [12] C. L. James and K. M. Reischel. Text input for mobile devices: Comparing model prediction to actual performance. In *Proceedings of the ACM Conference on Computer-Human Interaction*, pages 365–371, 2001.
- [13] D. J. Langendorf. Textware solution’s Fitaly keyboard v1.0 easing the burden of keyboard input. *WinCE Lair Review*, February 1998.
- [14] S. Mackenzie and W. Soukoreff. Text entry for mobile computing: Models and methods, theory and practice. *Human-Computer Interaction*, 17:147–198, 2002.
- [15] T. Masui. An efficient text input method for pen-based computers. In *Proceedings of the ACM Conference on Computer-Human Interaction*, pages 328–335, 1998.
- [16] G. Miller. The magical number seven plus or minus two; some limits on our capacity for processing information. *Psychological Review*, 63:81–97, 1956.
- [17] G. Nagy, L. Li, A. Samal, S. Seth, and Y. Xu. Cooperative text and line-art extraction from a topographic map. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition (ICDAR’99)*, pages 467–470, Bangalore, India, September 1999.
- [18] G. Nagy, L. Li, A. Samal, S. Seth, and Y. Xu. Integrated text and line-art extraction from a topographic map. *International Journal on Document Analysis and Recognition*, 2(4):177–185, June 2000.
- [19] J. P. Peters, C. Thillou, and S. Ferreira. Embedded reading device for blind people: a user-centred design. In *IEEE Emerging Technologies and Applications for Imagery Pattern Recognition (AIPR 2004)*, pages 217–222, 2004.
- [20] S. Srihari and Z. Shi. Forensic handwritten document retrieval system. In *First International Workshop on Document Image Analysis for Libraries (DIAL’04)*, pages 188–194, 2004.
- [21] K. Swan. FALCon: Evaluation of OCR and machine translation paradigms, August 1999. <http://www.arl.army.mil/seap/reports/kreport.pdf>.
- [22] J. Yang, X. Chen, J. Zhang, Y. Zhang, and A. Waibel. Automatic detection and translation of text from natural scenes. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP ’02)*, May 2002.
- [23] J. Zou. *Computer Assisted Visual InterActive Recognition*. PhD thesis, Rensselaer Polytechnic Institute, 2004.
- [24] J. Zou and G. Nagy. Evaluation of model-based interactive flower recognition. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 2, pages 311–314, 2004.

Arabic Document Analysis

Handwritten Arabic Word Spotting using the CEDARABIC Document Analysis System

Sargur Srihari, Harish Srinivasan, Pavithra Babu and Chetan Bhole

Center of Excellence for Document Analysis and Recognition (CEDAR)

University at Buffalo, State University of New York

Amherst, New York 14228

srihari @cedar.buffalo.edu

Abstract

An algorithm and a system for searching handwritten Arabic documents to locate key words is presented. Three main components of the system are a word segmenter, a shape based matcher for words and a search interface. The user types in a query in English within a search window, the system finds the equivalent Arabic word, e.g., by dictionary look-up, locates word images in an indexed (segmented) set of documents. A two-step approach is employed in performing the search: (1) prototype selection: the query is used to obtain a set of handwritten samples of that word from a known set of writers (these are the prototypes), and (2) word matching: the prototypes are used to spot each occurrence of those words in the indexed document database. A ranking is performed on the entire set of test word images— where the ranking criterion is a similarity score between each prototype word and the candidate words based on global word shape features. A database of 20,000 word images contained in 100 scanned handwritten Arabic documents written by 10 different writers was used to study retrieval performance. Using five writers for providing prototypes and the other five for testing, using manually segmented documents, 55% precision is obtained at 50% recall. Performance increases as more writers are used for training.

1 Introduction

Spotting handwritten words in documents written in the Latin alphabet has received considerable attention [1],[2], [3]. A systematic comparison of seven methods was made in [4] who showed that the highest precision was obtained by their method based on profiles and dynamic time warping (DTW). A method based on word shape was shown to perform better than the method based on DTW, both in terms of efficiency and effectiveness, in [5]. This paper discusses the performance of the word shape method presented in [5] when applied to handwrit-

ten Arabic documents.

The word spotting algorithm is implemented as a tool within an interactive software system known as CEDARABIC which is designed for the analysis of handwritten Arabic documents. Most functionalities of CEDARABIC are the same as that found in a predecessor system known as CEDAR-FOX which was developed for forensic examination by U.S. law enforcement [6]. Being a software branch of the CEDAR-FOX system CEDARABIC also has capabilities for image processing operations such as enhancement, thresholding, underline removal, etc., as well as for writer verification, writer identification and signature verification. The interfaces have been modified to be appropriate for analyzing Arabic language documents by an English-speaking user. The system includes a GUI for various functions including scanning, selecting a region of interest, feature computation, ground truthing (manual segmentation and entering word truth), searching of a word based on its English equivalent, etc. The primary focus of this paper is on the word-spotting tool of CEDARABIC—whose goal is to find each occurrence of a user-specified query word in a given set of documents.

In the CEDARABIC system the task of word spotting has two phases: indexing and search. In the indexing phase each document image in the database is indexed by segmenting into its component word images. The search phase has two parts: in the first part the typed in query is used to generate a set of query images corresponding to handwritten versions of it. In the second part each query image is matched against each indexed image to find the closest match.

The paper consists of the following sections: (i) scanned document collection used to design the system and and test its performance, (ii) user interface of CEDARABIC, (iii) word segmentation algorithm, (iv) word-shape matching algorithm, (v) conclusion and acknowledgment.

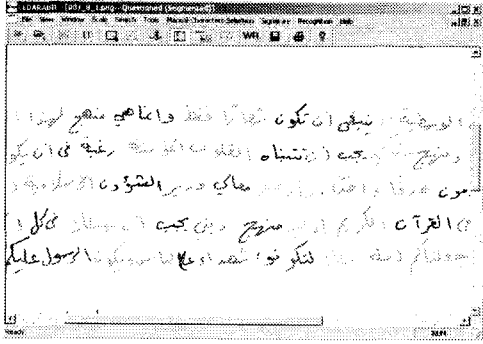


Figure 1: Sample “.arb” file opened with CEDARABIC. Each segmented word is colored differently from adjacent words.

2 Document Image Database

A document collection was prepared with 10 different writers, each contributing 10 different full page documents in handwritten Arabic. Each document comprises of approximately 150 – 200 words each, with a total of 20,000 word images in the entire database. The documents were scanned at a resolution of 300 dots per inch. This is the resolution to be used for optimal performance of the system. Figure 1 shows a sample scanned handwritten Arabic document written by writer 1.

For each of the 10 documents that were handwritten, a complete set of truth comprising of the alphabet sequence, meaning and the pronunciation of every word in that document was also given. The scanned handwritten document comprising of word images were mapped with the corresponding truth information given, and saved as a separate file with a “.arb” extension. The naming convention for the file is explained with this example. File 007_9.1.arb is the ninth document written by writer number 7. When some writers use more than one full page to transcribe a document, the last digit 1 or 2 identifies the page of the document. These “.arb” files have all the necessary information in them to perform word spotting using the tool that is described in the following section. The preparation of these “.arb” files involved careful manual word segmentation of each scanned document. The software tool described below also provides for a way to manually segment each word from a raw unprocessed document.

3 User Interface

Most of the functionalities of the CEDARABIC system, including word spotting, are the same as that found in a system for Latin script, known as CEDAR-FOX, which was developed for forensic examination by U.S. law enforcement [7] [8]. Being a modification of the CEDAR-FOX system for the analysis of handwritten documents in the Latin alphabet, CEDAR-FOX also has capabilities for image

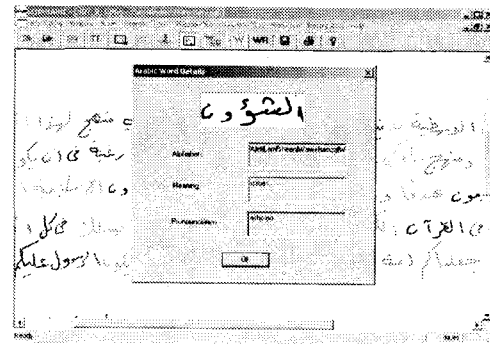


Figure 2: Truth for a particular word. This window pops up when a particular word image is right-clicked. The truth consists of three parts: the individual characters, its meaning in English and its Arabic pronunciation shown in English.

processing operations such as enhancement, thresholding, underline removal, etc., as well as for writer verification, writer identification and signature verification. The interfaces have been modified to be appropriate for analyzing Arabic language documents by an English-speaking user.

CEDARABIC has a GUI for analyzing and using the prepared “.arb” for performing word spotting and other information lookup tasks. A particular “.arb” file can be opened by selecting the “Q” or “K” option on the tool bar and Figure 1 shows a screen shot of CEDARABIC when a particular file is opened. By clicking on the tool bar icon for “Show Words”, the different words get marked with different colors thereby differentiating one Arabic word image from another.

Right-clicking on any word image pops up a dialog-box displaying the complete set of truths associated with that word. Figure 2 shows a screenshot when a particular word image was right-clicked. The Arabic alphabet sequence, the English equivalent meaning, and the pronunciation of each word can be viewed with this tool.

The query for word spotting is typed as English text- one that corresponds to the English equivalent meaning of the word to be searched for.

The first step in word spotting is to identify prototypes (template word images) corresponding to the Arabic word for the queried text. These template word images are obtained from documents which are set aside as training documents. For example, we can say documents pertaining to writers 1 through 5 can be used solely for the purpose of training and they provide for the template word images. The default set consists of writers 1 through 5. These training “.arb” should contain the complete truth for each word image, so as to pick up those that match the query text. The word images for those that matched serve as template word images to per-

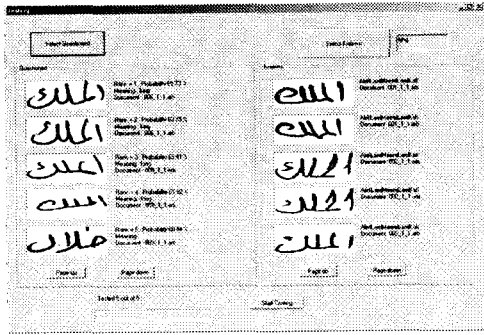


Figure 3: Word spotting results. The template word images on the right side, obtained in the first step, are used to spot the images shown on the left. The results are sorted in rank by probability. The images on the right are the selected prototypes and those on the left are the ranked set of test images.

form word spotting in other documents that do not have the truth. The matching words are automatically selected and the user is also given an option of selecting the ones that need to be used as template word images. Once this is done, upon clicking "Use Selected for Training" these images will be used to spot similar occurrences for words in other documents purely based on the shape of the word. The template word images serve as a knowledge base to learn the writing styles of the particular word and then use this to identify other similar words purely based on matching the image. A set of test documents can now be specified, and the tool will look for the occurrence of the particular word in every document specified and rank the retrieved words in the order of probability of match. Figure 3 shows the result of one such word spotting task performed. In this example the search is for the word "king" whose Arabic equivalent is pronounced as "al malik". The images on the right are the template word images and those on the left are the word images in order by rank.

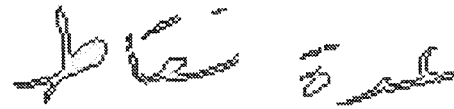
4 Word Segmentation

Automatic segmentation into words is based on taking several features on either side of a potential segmentation point and using a neural network for deciding whether or not the segmentation is between two distinct words. Some of the differences between the tasks of segmenting Arabic script and segmenting Latin script are: presence of multiple dots above and below the main body in Arabic and the absence of upper case letters at the beginning of sentences in Arabic.

The process of word segmentation begins with obtaining the set of connected components for each line in the document image. Figure 4 shows a region of a document image the connected components (exterior and interior contours) of a small section. The

متوسط / رسمي / رسمي / كلمة نقاط من جدول
 للسعي الرسمي المتوسخا تدرس با سبيلها

(a) Region of a document image

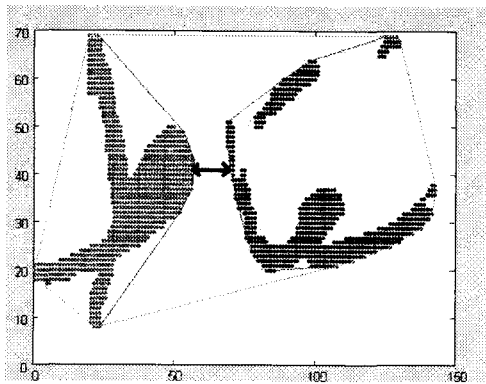


(b) Connected component exterior and interior contours

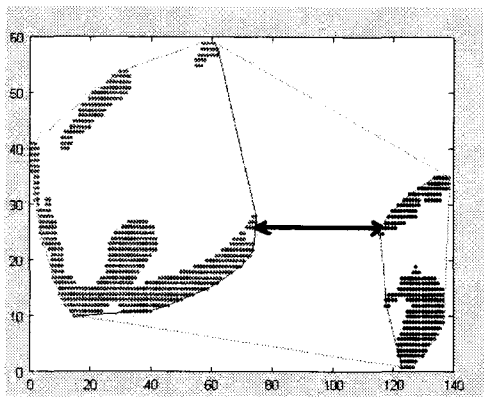
Figure 4: Connected components of a small section of the image.

interior contours or loops in a component are ignored for the purpose of word segmentation as they provide no information for this purpose. The connected components are grouped into clusters, by merging minor components such as dots above and below a major component. Also particular to Arabic, most words starts with the Arabic character "Alef". The presence of an "Alef" is a strong indicator that there may be a word gap between the pair of clusters. The height and width of the component are two parameters used to check if the component is the character "Alef". Figure 6 shows samples of the Arabic character Alef. Every pair of adjacent clusters are candidates for word gaps. 9 features are extracted for these pairs of clusters and a neural network is used to determine if the gap between the pair is a word gap. The 9 features are: width of the first cluster, width of second cluster, difference between the bounding box of the two clusters, flag set to 1 or 0 depending on the presence or absence of the Arabic character "Alef" in the first cluster, the same flag for the second cluster, number of components in the first cluster, number of components in the second cluster, minimum distance between the convex hulls enclosing the two clusters and the ratio between, the sum of the areas enclosed by the convex hulls of the individual clusters, to the total area inside the convex hull enclosing the clusters together. Figure 5 shows two pairs of adjacent clusters tested for word gaps.

A neural network was trained using these 9 features with feature vector labeled as to whether it is a word gap or not. This is similar to the neural network approach used for English postal addresses [9].



(a) Not word gap



(b) Word gap

Figure 5: Two pairs of adjacent clusters. Dotted lines indicate convex hulls around the individual clusters. Solid lines indicate convex hull around the two clusters taken together. (a) The gap between the convex hulls is truly not a word gap, and (b) The gap between the convex hulls is truly a word gap.

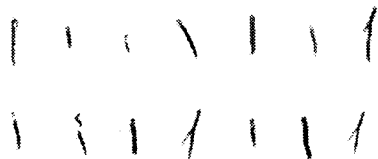


Figure 6: Samples of Arabic character "Alef". The height and width are two parameters that are used to detect the presence of "Alef" in the clusters.

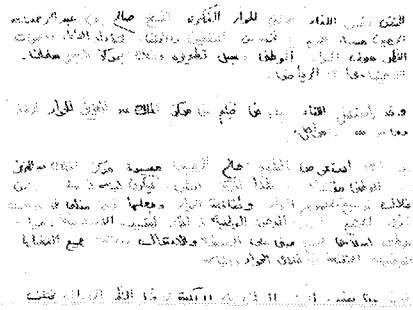


Figure 7: A document segmented into words by the neural network. Figure 8(a) 8(b) 8(c) and 8(d) show the first four clusters of the document for the first line from the left and the classification result.

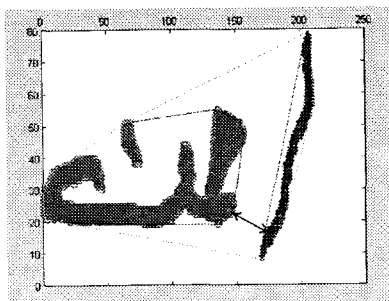
Table 1: NN Features

Fig	f1	f2	f3	f4	f5	f6	f7	f8	f9	NN
8(a)	156	42	12	0	0	2	1	24.59	0.58	-0.75
8(b)	42	13	7	0	1	1	1	8.00	0.52	-0.99
8(c)	13	84	14	1	0	1	2	24.75	0.40	0.91
8(d)	84	82	51	0	0	2	2	51.00	0.56	0.80

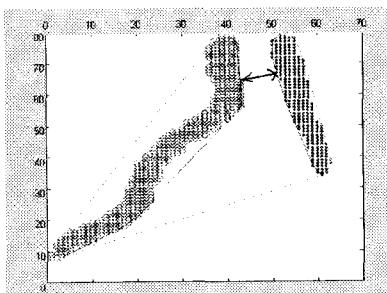
However the features are different. Figure 7 shows a document segmented into words by the neural network. Figure 8 shows the first 4 pairs of clusters whose features are extracted and fed to the neural network. These 4 pairs of clusters correspond to the words starting from the left on the first line of the document in figure 7. Table 1 shows the 9 features for these pairs respectively and the last column shows the output of the neural network and table 2 shows what each feature is. A positive output from the neural network indicates a presence of word gap and a negative output indicates the absence of a word gap. From the table 1, it is evident that though the distance between the clusters for pair 1 and pair 3 are almost the same, the neural network gives the correct classification. This is due to presence of "Alef" that strongly indicates the presence of a word gap.

Table 2: Features

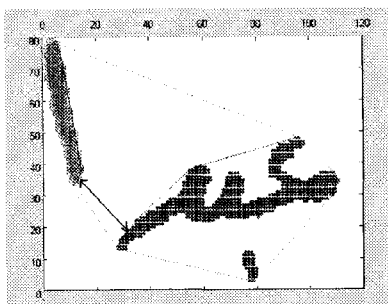
Number	Feature
1	Width of 1 st cluster
2	Width of 2 nd cluster
3	Bounding box gap between clusters
4	Alef flag for 1 st cluster
5	Alef flag for 2 nd cluster
6	No of components 1 st cluster
7	No of components 2 nd cluster
8	Min. Convex hull distance
9	Ratio of sum of cluster areas to total area



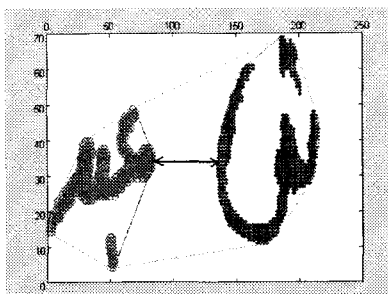
(a) Not word gap



(b) Not word gap



(c) Word gap



(d) Word gap

Figure 8: Four pairs of adjacent clusters. Note that the distance between the clusters in (a) and (c) are the same (the scale in the plot are different). The neural network still classifies (c) as a word gap due to the presence of the "Alef".

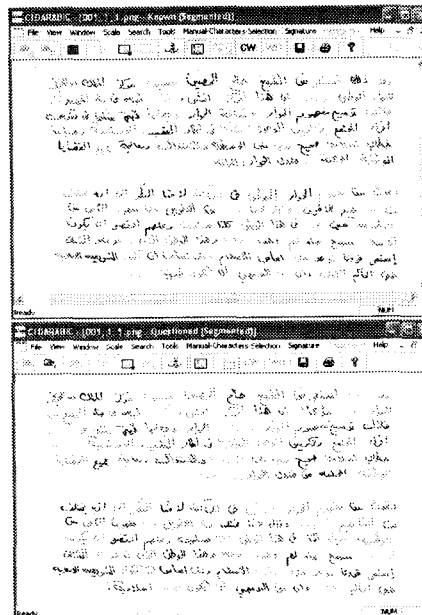


Figure 9: Example of a reasonably well-segmented Arabic document by automatic segmenter designed for English. The truth is shown on the right screen-shot and the result of the automatic word segmenter is shown on the left. In this case 86% of the automatically segmented words are correct.

4.1 Word Segmentation Performance

An example of a reasonably well-segmented Arabic image by the automatic segmenter, together with the perfectly segmented image (truth), are shown in Figure 9. There are 130 words in the truth shown on the right-hand side. The result of the automatic segmenter, shown on the left, has the following error counts: (i) 9 words have been split into two parts (9 errors), (ii) 3 adjacent pairs of words have been merged (6 errors) and (iii) one set of three words have been merged (3 errors). Since there are a total of 18 errors we can say that 86% of the word segmentations are correct.

An example of a poorly segmented image together with its truth are shown in Figure 10. In this case there are 120 words in the truth shown on the right. the error counts of the automatically segmented document on the left are as follows: (i) 17 words have been split (17 errors), (ii) 18 adjacent words have been merged (36 errors), (iii) 3 merges of three adjacent words (9 errors), and (iv) one merge of four adjacent words (4 errors). Since there are 66 errors we can say that 45% of the word segmentations are correct.

Overall performance over ten writers writing 10 documents each, when the lines are segmented correctly, is about 60% using a set of seven segmentation features. A more complex set of features is expected to yield a higher level of performance.

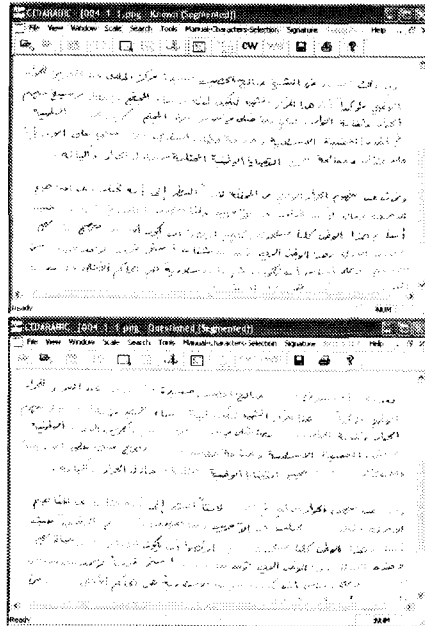


Figure 10: Example of a poorly segmented Arabic document by automatic English segmenter. Truth is on the right and the result of automatic segmentation is on the left. In this case 45% of the segmented words are correct.

5 Word Shape Matching

The word segmentation is an indexing step before using the word shape matching for word retrieval. A two-step approach is employed in performing the search: (1) prototype selection: the query (English text) is used to obtain a set of handwritten samples of that word from a known set of writers (these are the prototypes), and (2) word matching: the prototypes are used to spot each occurrence of those words in the indexed document database. A ranking is performed on the entire set of test word images—where the ranking criterion is the mean similarity score between prototype words and the candidate word based on global word shape features.

5.1 Prototype Selection

Prototypes which are handwritten samples of a word are obtained from an indexed (segmented) set of documents. These indexed documents contain the truth (English equivalent) for every word image. Synonymous words if present in the truth are also used to obtain the prototypes. Hence queries such as “country” will result in selecting prototypes that have been truthed as “country” or “nation” etc... A dynamic programming Edit Distance algorithm is used to match the query text with the indexed word image’s truth. Those with distance as zero are automatically selected as prototypes. Others can be selected manually.

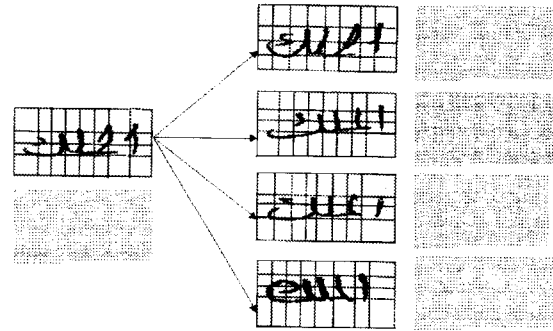


Figure 11: One query word image (left) matched against 4 selected prototype images. The 1024 binary GSC features are shown next to the images.

5.2 Word Matching

The word matching algorithm uses a set of 1024 binary features for the word images. These binary features are compared using the correlation similarity measure 5.2.1 to obtain a similarity value between 0 and 1. This similarity score represents the extent of match between two word images. The smaller the score, the better is the match. For word spotting, every word image in the test set of documents are compared with every selected prototype and a distribution of similarity values is obtained. The distribution of similarity values is replaced by its arithmetic mean. Now every word is sorted in rank in accordance with this final mean score. Figure 5.2 shows an example query word image compared with the a set of 4 selected prototypes.

5.2.1 Similarity measure

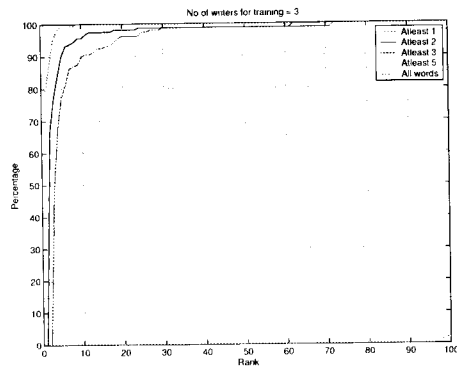
The method of measuring the similarity or distance between two binary vectors is essential. The correlation distance performed best for GSC binary features [10] which is defined for two binary vectors X and Y , as in equation 5.2.1

$$d(X, Y) = \frac{1}{2} \left(1 - \frac{s_{11}s_{00} - s_{10}s_{01}}{[(s_{10} + s_{11})(s_{01} + s_{00})(s_{11} + s_{01})(s_{00} + s_{10})]^{\frac{1}{2}}} \right)$$

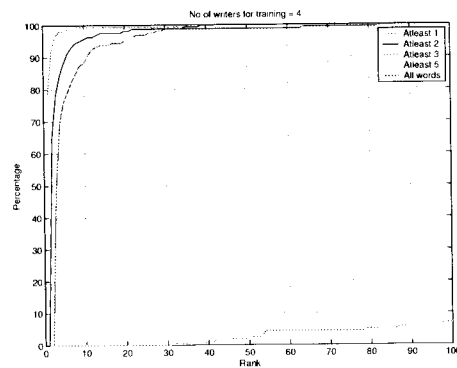
where s_{ij} represent the number of corresponding bits of X and Y that have values i and j .

5.3 Word-Matching Performance

The performance of the word spotter was evaluated using manually segmented Arabic documents. All experiments and results were averaged over 150 queries. For each query, a certain set of documents are used as training to provide for the template word images and rest of the documents were used for testing. Figure 12 shows the percentage of times when correct word images were retrieved within the top ranks scaled on the x-axis. More than one correct word image does exist in the test set. For example, if the word “king” occurs in document 1 twice, and we use writers 1 through 6 for training, then



(a) Recall with 3 writers for training.

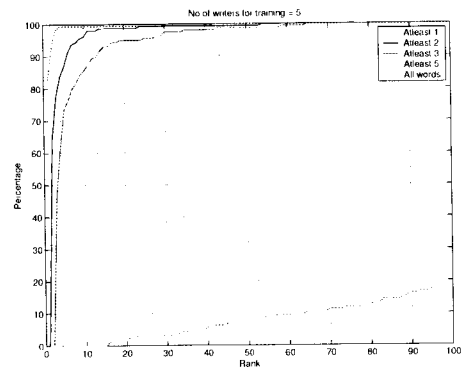


(b) Recall with 4 writers for training.

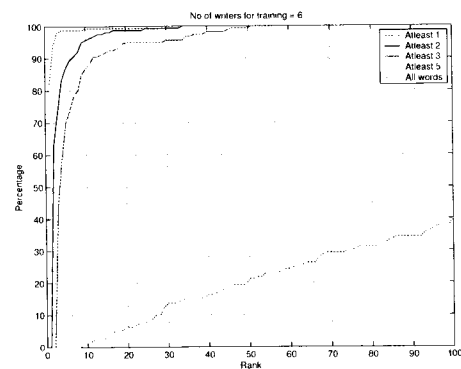
Figure 12: Retrieval performance of the system. All results were averaged over 150 queries. The styles of 3 and 5 writers were used for training (by using their documents) to provide template word images. The rest were used for testing. The different curves report on the percentage of times the word images were retrieved within the ranks scaled on the x-axis.

we have $6 * 2 = 12$ template word images to perform word spotting. The remaining test set contains $(10 - 6) * 2 = 8$ correct matches in the test set. The top curve of the figure shows that at least one correct word is retrieved 90% of the time within the top 12 ranks. The other curves show the same when at least 2 and 3 correct words are retrieved. Similar plots can be obtained when different numbers of writers are used for training. These plots are shown in Figs. 12 13 and 14 where three, five and seven writers were used for training.

Performance of word spotting can be specified in terms of precision and recall. If n_1 is the number of relevant words in the database, n_2 is the number of words retrieved, and n_3 is number of relevant words among those that are retrieved then $Precision = n_3/n_2$ and $Recall = n_3/n_1$. Precision-Recall curves corresponding to using three, five and

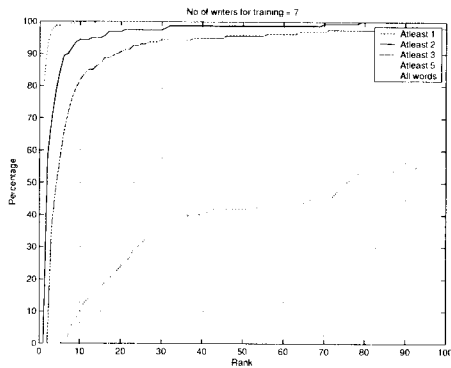


(a) Recall with 5 writers for training.

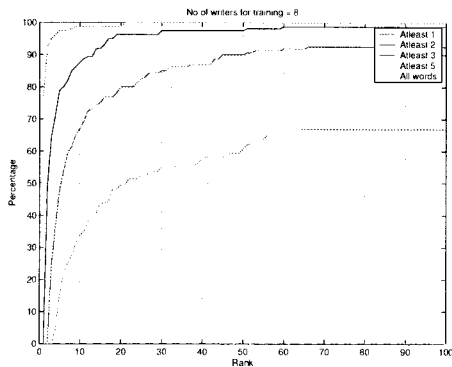


(b) Recall with 6 writers for training.

Figure 13: Retrieval performance of the system. All results were averaged over 150 queries. The styles of 5 and 6 writers were used for training (by using their documents) to provide template word images. The rest were used for testing. The different curves report on the percentage of times the word images were retrieved within the ranks scaled on the x-axis.

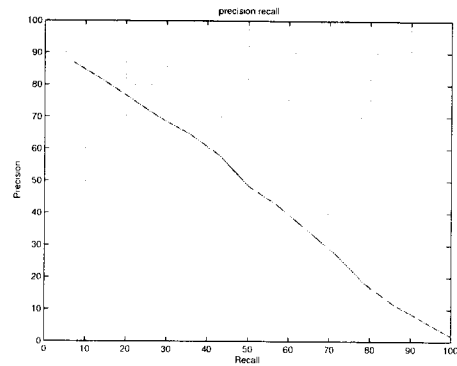


(a) Recall with 7 writers for training.

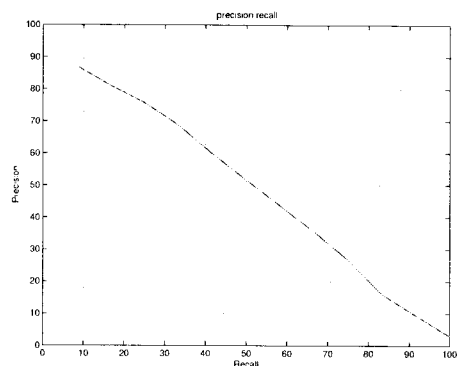


(b) Recall with 8 writers for training.

Figure 14: Retrieval performance of the system. All results were averaged over 150 queries. The styles of 7 and 8 writers were used for training (by using their documents) to provide template word images. The rest were used for testing. The different curves report on the percentage of times the word images were retrieved within the ranks scaled on the x-axis. Results show improved performance over the case when five styles were used as shown in Figure 12 and 13.



(a) Precision Recall with 3 writers for training.



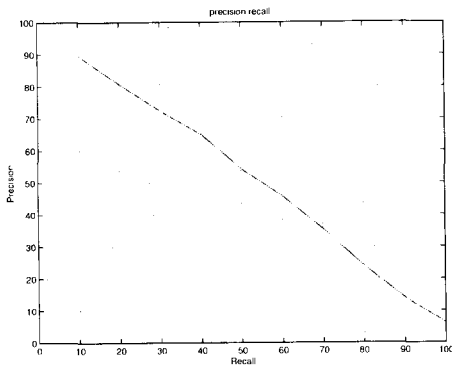
(b) Precision Recall with 4 writers for training.

Figure 15: Precision-Recall curves where the styles of 3 and 4 writers were used for training (by using their documents) to provide template word images.

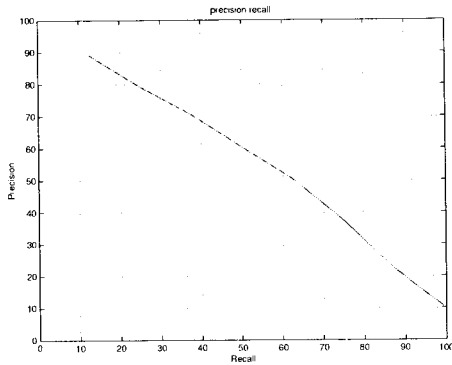
seven writer styles are shown in Figures 15 , 16 and 17 respectively. Finally figure 18 compares the precision at a particular recall with varying number of writers used for training. As the number of writers is increased, the precision at a given recall increases.

6 Conclusion

A technique for Arabic word image spotting in previously segmented images and that of automatic Arabic word segmentation was discussed. A two-step approach involving (i) Prototype selection and (ii) Word matching were described. Performance of the system is very promising with increased recall and better precision as the number of writers in the training set is increased. A precision of 70% is achieved at a recall of 50% when 8 writers were used for training. The CEDARABIC system described as the user interface provides tools for (i) manual word segmentation, (ii) information look-up on truth of every word, and (iii) interface to perform word spotting. A set

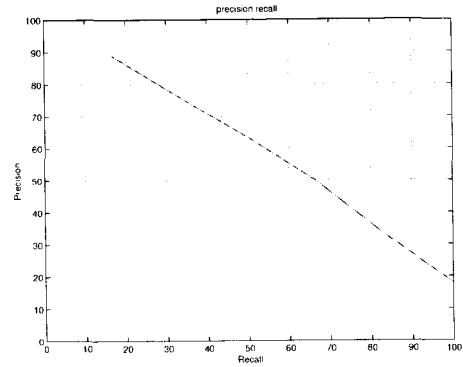


(a) Precision Recall with 5 writers for training.

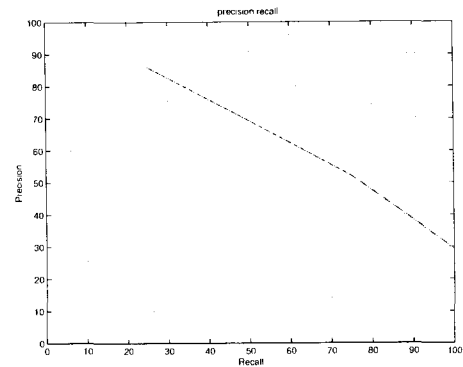


(b) Precision Recall with 6 writers for training.

Figure 16: Precision-Recall curves where the styles of 5 and 6 writers were used for training (by using their documents) to provide template word images.



(a) Precision Recall with 7 writers for training.



(b) Precision Recall with 8 writers for training.

Figure 17: Precision-Recall curves where the styles of 7 and 8 writers were used for training (by using their documents) to provide template word images. Results show an improvement over the case when five styles were used as shown in Figure 15 and 15.

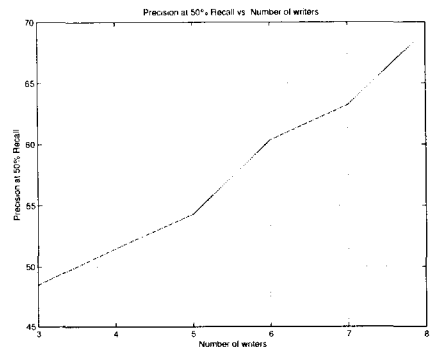


Figure 18: Precision at 50% recall is plotted against different numbers of writers used for training. As more number of writers are used for training, the precision at a given recall increases.

of more complicated features tuned for Arabic will result in better automatic word segmentation performance and it will be a part of future research.

7 Acknowledgment

The authors wish to thank Chen Huang, Vivek Shah, Manivannan Arivazhagan and Thanh Nguyen for their assistance.

References

- [1] S. Kuo and O. Agazzi, "Keyword spotting in poorly printed documents using 2-d hidden markov models," in *IEEE Trans. Pattern Analysis and Machine Intelligence*, **16**, pp. 842-848, 1994.
- [2] M. Burl and P. Perona, "Using hierarchical shape models to spot keywords in cursive handwriting," in *IEEE-CS Conference on Computer Vision and Pattern Recognition*, **June 23-28**, pp. 535-540, 1998.
- [3] A. Kolz, J. Alspecter, M. Augusteijn, R. Carlson, and G. V. Popescu, "A line-oriented approach to word spotting in handwritten documents," in *Pattern Analysis and Applications*, **2(3)**, pp. 153-168, 2000.
- [4] R. Manmatha and T. M. Rath, "Indexing of handwritten historical documents-recent progress," in *Symposium on Document Image Understanding Technology (SDIUT)*, pp. 77-85, 2003.
- [5] B. Zhang, S. N. Srihari, and C. Huang, "Word image retrieval using binary features," in *Document Recognition and Retrieval XI*, **SPIE Vol. 5296**, pp. 45-53, 2004.
- [6] S. N. Srihari, B. Zhang, C. Tomai, S. Lee, and Y.-C. Shin, "A system for handwriting matching and recognition," in *Symposium on Document Image Understanding Technology Greenbelt, MD*, April, 2003.
- [7] S. N. Srihari, S.-H. Cha, H. Arora, and S. Lee, "Individuality of handwriting," in *Journal of Forensic Sciences*, **47(4)**, pp. 856-872, 2002.
- [8] S. N. Srihari, B. Zhang, C. Tomai, S. Lee, Z. Shi, and Y. C. Shin, "A system for handwriting matching and recognition," in *Proceedings of the Symposium on Document Image Understanding Technology (SDIUT 03)*, *Greenbelt, MD*, pp. 67-75, 2003.
- [9] G. Kim, V. Govindaraju, and S. N. Srihari, "A segmentation and recognition strategy for handwritten phrases," in *International Conference on Pattern Recognition, ICPR-13*, pp. 510-514, 1996.
- [10] B. Zhang and S. N. Srihari, "Binary vector dissimilarity measures for handwriting identification," *Proceedings of the SPIE, Document Recognition and Retrieval*, pp. 155-166, 2003.

Word spotting in Arabic script documents

A. Lawrence Spitz and Jim Yaghi

DocRec Ltd
34 Strathaven Place, Atawhai, Nelson 7001, New Zealand
spitz@docrec.com, jim@docrec.com

Abstract

Several applications of processing of document text based on character shape codes have been demonstrated for documents in English and other languages set in Roman script, but ability to deal with Arabic-script documents has not yet reached the same level of sophistication.

Arabic script does not naturally divide into characters in the image domain complicating the design of shape code sets. Many Arabic fonts are quite ornate adding complexity to the shape classification process and the Arabic language is agglutinative and highly inflected and thus requires a huge lexicon of surface forms.

These attributes combine to make reconstruction of Arabic considerably more complex than has been shown for Roman-script documents.

We are developing a technique for spotting the occurrences of particular words of interest in Arabic script document images.

1 Introduction

A character shape representation has been shown to support many applications such as text reconstruction[1], postal address recognition[2], duplicate detection[3], language identification[4] and topic identification[5] amongst others. Most of the work has been applied to Roman-script documents, though there have been reports of successful application in both Thai [6] and Japanese [7] documents.

In Section 2 we describe some of the attributes of Arabic text which must be taken into consideration in shape coding.

Section 3 describes the image processing necessary to perform Arabic Shape Coding. Section 4 describes the shape representation for Arabic script. Section 5 will deal with issues of representation ambiguity.

Section 6 will describe our word-spotting experiments and Section 7 discusses current status and future plans.

2 Arabic Document Image Attributes

2.1 Typography

Let us define a *grapheme* as being a connected sequence of characters within or comprising a word. The Arabic alphabet has 28 letters, 22 of which have four forms depending on whether the letter is in isolation, in the initial, medial or terminal position of a grapheme. The remaining 6 letters have only two forms, isolated and terminal. The sequence of characters within a word is connected until and unless one of these six letters is encountered, thereby ending the grapheme. Spaces between graphemes are smaller than spaces between words.

Arabic characters often contain ancillary marks such as hamza **أ** and nuqta **ق**. Some Arabic characters share skeletal forms and therefore are distinguished only by the number and position of these nuqta **ب ت ث**.

Justification in Arabic script is accomplished using *keshide*, or horizontal stretchiness of character images, obviating the need for hyphenation. Thus, words are never fragmented across text line boundaries.

2.2 Fonts

There is a wide variety of Arabic fonts in use ranging from simple to very ornate. Fortunately most body text is set in one of a few simple fonts.

2.3 Language and Lexicography

In agglutinating languages, words may be comprised of several morphemes, but are easily divided into the components that make them up. These components are usually one root and several affixes. Each affix typically has one meaning and represents

only one grammatical category. On the other hand, words in derivational languages do consist of several morphemes but their affixes represent several grammatical categories at the same time and the affixes are not easily detachable because they interdigitate with the root. Some languages have a high amount of agglutination or fusion, allowing for a large number of morpheme combinations. Latin languages are primarily fusional. Arabic and English are both fusional and agglutinating, although Arabic is more fusional.

Arabic words can take several forms depending on their position in the sentence. Verbs change form depending on (1) the subject's number (singular, dual, plural), gender (masculine, feminine), and person (first, second, third); (2) the verb's tense (past, present, imperative); (3) voice (passive, active); and (4) mode (indicative, subjunctive, jussive). The English equivalent of Arabic's indicative mode of the verb study is either study or studies as in 'they study', 'she studies', and the subjunctive mode equivalent is English's non-finite verb study as in 'I recommend that she study hard', etc. Whilst the English representation of the verb *study* only uses two forms, Arabic uses a large number of word forms, as can be seen below. The Arabic equivalent of the verb *study* takes a large array of orthographic representations depending on its syntactic context. This example shows the active voice, present tense verb inflected for different subjects. Other word forms can be generated from the remaining tense and voice combinations. A second pass of word form modification may take place when adding object pronominal suffixes, conjunction particles, and verb intensifiers. The latter modifications may be primarily agglutinating but they do add a second level of complexity to word forms.

Table 1: Tense: Present, Voice: Active for Arabic

Pronoun	Indicative	Subjunctive	Jussive
He	yadorusu	yadorusa	yadoruso
She	tadorusu	tadorusa	tadoruso
You (masc.)	tadorusu	tadorusa	tadoruso
You (fem.)	tadorusyna	tadorusy	tadorusy
I	>adorusu	>adorusa	>adoruso
They (dual, masc.)	yadorusAni	yadorusA	yadorusA
They (dual, fem.)	tadorusAni	tadorusA	tadorusA
You (dual)	tadorusAni	tadorusA	tadorusA
We	nadorusu	nadorusa	nadoruso
They(plural,masc.)	yadoruswna	yadoruswA	yadoruswA
They (plural, fem.)	yadorusona	yadorusona	yadorusona
You (plural, masc.)	tadoruswna	tadoruswA	tadoruswA
You (plural, fem.)	tadorusona	tadorusona	tadorusona

This variation in verb forms causes a single verb to have hundreds of orthographic representations. The same type of complexity applies to nouns, where each noun is marked for gender, number, definiteness, and grammatical case.

In addition, the fact that the language typically indicates the consonants, semi-vowels, and vowels of words and makes the marking of short vowels optional adds yet another level of difficulty for graphic recognition. Arabic text may have some, all, or none of the short vowel diacritics marked. In most cases, diacritics are omitted entirely and added only when the context may cause confusion in meaning. Fortunately, many of these diacritics are small marks that appear above or under a base grapheme; hence, they can be ignored completely. It might be wise to eliminate short vowel diacritics altogether in order to reduce the size of lexicon. Even then, the size of the lexicon is extraordinary.

It is these derivational properties of the morphology and the inflectional nature of the syntax which make it difficult to create an encompassing lexicon of the language. Using a small corpus is unlikely to facilitate finding examples of the majority of words in the majority of contexts where they occur.

3 Image processing

Most of the image processing is the same as for Roman text. See [9]. Text lines are processed from left to right but at the end of the line the shape code sequences are reversed.

3.1 Text-line fiducials

In shape coding Roman script, characters are classified by the spaces that they occupy: the ascender zone above the x-line, the zone between the baseline and the x-line, and the descender zone below the baseline.

Shape coding Arabic script requires different fiducials largely because the concept of x-height does not apply well to Arabic script. Nevertheless for want of better terms we have continued to use x-line to define the vertical position that delimits the ascender zone from the zone immediately adjacent to the baseline. Likewise x-height is the distance from the baseline to the x-line and x-zone is the zone between those two fiducials.

3.2 Quanta

In Arabic, joined character pairs¹ are the rule rather than the exception since in most cases non-terminal characters within a word are connected to the following character.

Thus we were presented with three options for segmenting Arabic text. In each case we generate a sequence of Arabic Shape Codes (ASCs) for each quantum encoded.

3.2.1 Grapheme segmentation

In this instance we would generate a shape code for each connected sequence of letters. There are an average of 3.6 and a maximum of 12 letters per grapheme in the lexicon. This requires a fairly large shape code set to represent the complexity contained in the grapheme, but is the easiest from the standpoint of image processing since all that needs to be distinguished before encoding is the difference between intra-word (grapheme) spacing and inter-word spacing.

3.2.2 Character segmentation

To encode actual characters one-to-one, accurate segmentation must be performed. All joining character pairs do so with a horizontal segment at the baseline, but unfortunately some characters have baseline-aligned horizontal segments within them. Thus character segmentation becomes tricky.

The benefit of true character coding is large however since the transformation from the image domain to the lexical (character) domain is trivial.

3.2.3 Over-segmentation

In this instance, drawing from the experience of those engaged in (Roman) handwriting recognition[8], we divide the image not only at grapheme spaces but at all instances of horizontal segments at the baseline. This will clearly isolate all characters but will also break some characters into multiple segments.

In order to find segments, we search a line from one end to the other delimiting graphemes when the distance between the highest point of the current pixel column and the lowest pixel column exceeds the average stroke-height by a margin. The margin is decided by how rapidly the pixel column height increases. Following this approach allows us to split graphemes at even the smallest teeth.

We over-segment graphemes and simply search for *teeth* or bumps above the connector between letters. Instead of considering the letter **ب** in grapheme-medial position to be made up of one x-height character, we count it as two x-height segments. So the word **كتب** will be considered to be made up of one

ascender segment **ك**, followed by two x-height segments **ت** and the word **بصر** is made up of three x-height segments followed by a descender. The letter **ص** here was counted as two segments, the curved part was one while its tooth was the second.

If 100% accurate segmentation of any of the three types cited, could be performed there would be no need for any indication in the shape coded regime of grapheme breaks within a word. However we cannot count on perfect performance of any of these algorithms and so in particular to support over-segmentation code streams we must insert *end-of-grapheme* indicators to allow re-alignment in the case of errors.

3.3 Word boundaries

In Roman script, a single line of text contains a relatively large number of small inter-character spaces, and a smaller number of wider inter-word spaces. The distribution of space width shows two peaks indicating the modal widths of inter- and intra-word gaps. Because of the connected nature of Arabic, there are many fewer spaces per line. Although grapheme spaces are still more numerous than word spaces, grapheme spaces are often not markedly different from word spaces.

Arabic text has a large number of auxiliary marks above, below, and sometimes between letters. These marks may be diacritics, hamzas, or niqat (dots). Because diacritics in Arabic are not consistently placed on all letters, we do not currently handle them. A majority of business text and news uses few diacritic markings if any at all. However, the hamzas and niqat are extremely important for distinguishing many letters from one another as some letters share the same basic skeletal form and only differ in number or position of niqat. We assume all hamzas and niqat are placed somewhere within the width of the region spanned by the character's skeletal. These auxiliary markers appear either above or below the base-line, so we characterize each segment with a property indicating the position of its auxiliary marks if there are any. Letters such as **ش** have three niqat placed very close to one another in a triangular form above the letter's teeth. These niqat may or may not touch, and they may be offset so that they are in the region of either of the two segments that compose its teeth, but their shape is guaranteed to be triangular. We simply associate the markers with one of the segments of the letter and at the word recognition stage we use algorithms that allow these different possibilities to match the same letter.

1. Not to be confused with ligature characters (e.g. lam-alef) that represent character pairs with an entirely distinct form.

4 Shape classification

Having settled on a shape code representation based on over-segmentation, we define a four bit shape code for each segment thus encoded. Two bits define whether the skeletal of the character occupies the ascender, x-height and descender zones and one bit each to define the location of auxiliary marks above or below the skeletal.

Word Shape Tokens (WSTs) are shape code sequences delimited by word separators.



Figure 1: A few words of Arabic text showing the shape codes for each segment

5 Representation ambiguity

If we had a perfect OCR there would be no ambiguity. That is for each word shape code there would be one, and only one, corresponding entry in the surface form lexicon. Shape coding is a computationally efficient but lossy process and therefore, even if done perfectly, there is an ambiguity of representation since one shape code may represent more than one grapheme, character or segment.

We measure ambiguity in several ways, two of which yield a concise metric for the expected utility: average number of surface form words per WST and the proportion of WSTs that represent exactly one surface form word.

The actual ambiguity is a function of lexicon size and diversity (generality) [9] and the characteristics of the shape code set.

Our initial experiments based on coding entire graphemes with a single shape code yielded unacceptable results with an average of 67 words encoded by each WST

Our next experiment was to attempt character segmentation which resulted in a dramatic reduction in representation ambiguity but was too highly sensitive to incorrect character segmentation.

We have finally settled on an over-segmentation of the graphemes. We believe the results, shown in Table 2 below, will be sufficiently good to support a number of shape code based applications. This shape representation leads to a lexicon structure as shown in Table 3

Table 2: Ambiguity of various Arabic shape code representations with English, French and German for comparison.

	Lexicon Size	Shape code set size	Ambiguity	Singleton ratio
Arabic				
Grapheme	24320070	16	67.22	0.19
Character	24320070	12	3.78	0.47
Segment	24320070	12	2.46	0.59
Roman				
English	246906	5	1.40	0.84
French	514637	5	1.73	0.76
German	316035	6	1.51	0.75

Table 3: Snippet of lexicon indexed by word shape token

Word Shape Token	Word 1	Word 2	Word 3
0000 0001 0100 0000 0000 0000 0010	مناسب suitable	متكسب gainfully employed	
0000 0100 0000 0010	يكعب make cubic	يلعب he jests	يلعب he plays
0000 0000 0000 0000 0001 1000 0100 0100 0000 0101	مستهلكك customer of you		
0000 0000 0000 0000 0001 1000 1000 0000 0000	مستورد importer		
0000 0000 0000 0001 0000 0000 0000 0001 0000 0001 0000 0000 1001	ستستمنحين you will be given a grant	ستستمنعين you will enjoy yourself	

6 Word spotting

We have previously demonstrated the efficacy of character shape coding of Roman script for a word-spotting application [10]. We now adapt this technique to Arabic script.

We used as a database 100 document images of synthesized news articles. We selected 20 nouns from current news feeds to be used as search terms in this experiment. Table 4 shows the various representations of the search terms along with their frequency of detection in . A document was considered to be found if the search term is found in the document.

The first stage was to check whether we achieved 100% precision and recall based on the Unicode search term applied to Unicode news articles.

The second stage was designed to demonstrate the potential for Arabic Shape Coding and for the particular algorithmic transliteration that we have developed. Transliterations of the Unicode search

Table 4: Different representations of search terms and their frequency of occurrence in the database, trasliterated truth and image-based Arabic Shape Codes.

Arabic	Buckwalter	ASC	Approximate translation	Unicode	Shape-coded transliterated Unicode	Shape-coded document image
حقوق	Hqwq	0189	rights	45	45	44
استخدام	AstxdAm	4000110048	usage	52	52	49
لمجلس	lmjls	4024000	for the council	51	52	52
تحقيقات	AETyAl	4112448	arrest	60	63	63
اتفاقية	AtfAqyp	4114125	contract/ agreement	38	38	37
لندن	lndn	41009	London	48	48	48
غرينتش	grynt\$	18211010	Greenwich	38	38	35
الجميع	AljmyE	442028	everyone	39	42	38
النشر	Aln\$r	4410108	publishing	38	38	32
المعرفة	AlmErfp	4400815	familiarity/ knowledge	39	39	33
باكستان	bAkstAn	244000149	Pakistan	39	39	33
الصحافة	AlSHAfp	44000415	journal	36	39	26
جولة	jwtlp	2845	tour	35	35	30
قادة	qAdp	14005	leadership	44	60	58
واشنطن	wAshnTn	84010149	Washington	36	36	34
الغربيين	Algrbyyn	44182229	Westerners	34	34	31
السلطة	AlslTp	44000445	ruling power	24	29	26
تحقيقات	tHqyqAt	10121410	interrogations/ investigations	13	13	13
الطاقة	AlTAqp	444415	energy	20	25	24
محكمة	mHkmp	00405	court	10	10	10

terms into WSTs were probed for in the transliterations of the Unicode news articles. In an error-free environment, the expectation is that recall would be

100%. Precision would be somewhat reduced due to the ambiguity introduced by the shape coding process.

The third stage of the experiment is to evaluate performance based on transliterated search terms and synthesized document images.

Precision and recall measures for each stage are shown in Table 5.

Table 5: Results

	Unicode	Shape-code transliterated Unicode	Shape coded document image
Precision	100	95.4	93.8
Recall	100	100	97.0

From these data we can see that the shape coding process introduces a very small amount of ambiguity and the process of deriving shape codes from images introduces noise that slightly degrades both precision and recall.

A fourth step, not yet realized, is to degrade the document images in various ways and to re-run stage 3.

7 Status and plans

We believe we have developed a useful tool for spotting the occurrences of specific search terms in document images.

What we described here is a core technology around which we believe, based on prior experience with Roman character shape coding and a demonstrated favorable level of shape coding ambiguity for Arabic, will support a wide range of applications.

We have shown reasonable results in terms of precision and recall on Arabic script documents at a low computational burden and with robustness to image quality degradation characteristic to shape coding as opposed to OCR.

Other applications, as alluded to in the Introduction, will follow.

Acknowledgments

We would like to acknowledge the willing assistance we have received during the process of this development from Chuck Bigelow, Kareem Darwish, David Doermann, Ali Ferghaly, Kamal Mansour, Suzanne Nesbit, Douglas Oard, John Trenkle and Sane Yagi.

We would also like to acknowledge partial financial support for this project from the New Zealand Foundation for Research Science and Technology.

References

- [1] A. Lawrence Spitz, "Progress in document reconstruction", *International Conference on Pattern Recognition*, Quebec City, pp 464-467, 2002.
- [2] A. Lawrence Spitz, "Applications of a feeble text image classifier", *Image and Vision computing, New Zealand*, Auckland, pp 331-334, 2002.
- [3] Daniel P. Lopresti, A. Lawrence Spitz, "Comparing the utility of optical character recognition and character shape coding in duplicate document detection", *Document Analysis Systems*, pp 439-450, 2000.
- [4] A. Lawrence Spitz, "Determination of the script and language content of document images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 3, pp 235-245, 1997.
- [5] A. Lawrence Spitz, Arman Maghbouleh, "Text categorization using character shape codes", *SPIE Symposium on Electronic Imaging Science and Technology*, pp 174-181, 2000.
- [6] Doug Cooper, "How to read less and know more: Approximate OCR for Thai", *SIGIR*, Philadelphia, pp 216-225, 1997.
- [7] Masaharu Ozaki, Katsuhiko Itonori, "A fast Japanese word extraction with classification to similarly-shaped character categories and morphological analysis", *Document Analysis Systems*, Nagano, pp 316-325, 1998
- [8] Horst Bunke, "Recognition of cursive handwriting - past, present and future", *International Conference on Document Analysis and Recognition*, Edinburgh, pp 448-459, 2003.
- [9] A. Lawrence Spitz, "Moby Dick meets GEOCR: Lexical considerations in word recognition", *International Conference on Document Analysis and Recognition*, Ulm, Germany, pp 221-226, 1997.
- [10] A. Lawrence Spitz, "Using Character Shape Codes for Word Spotting in Document Images", *IAPR Workshop on Syntactic and Structural Pattern Recognition*, Narariya, Israel, pp 1-8, 1994.

Challenges in Adapting Machine-Print Arabic OCR for Handwriting

Steven G. Schlosser Robert C. Vogt

NovoDynamics, Inc.
123 N. Ashley St. STE 210
Ann Arbor, MI 48104
steve@novodynamics.com

Abstract

The creation of a handwritten Arabic recognition system capable of processing degraded pages presents many technical challenges. Some of those challenges are particular to the Arabic language and some to handwriting recognition in general. Other challenges are common to both machine-print and handwritten recognition because handwritten Arabic often resembles machine-print Arabic. Therefore, an effective approach to the development of a handwritten Arabic OCR may be to start with a machine-print Arabic OCR.

NovoDynamics has undertaken the task of adapting its high-performance machine-print Arabic OCR for Arabic handwriting. This paper describes the motivation for pursuing this particular approach as well as the initial results obtained. Performance metrics for early versions of an Arabic handwritten system applied to degraded pages are reviewed, and prospects for future progress described. It is shown that NovoDynamics' machine-print Arabic OCR is a solid foundation for Arabic handwriting recognition and that it has characteristics that make it an attractive candidate for adaptation to other languages as well.

1 Introduction

NovoDynamics' machine-print Arabic Character Recognition System (ACRS) and supporting tools have been developed to tackle the inherent difficulties of performing character recognition on low-quality document images. Because machine-printed Arabic is script-based, the underlying algorithms in ACRS are readily adapted to handwritten documents. The system also incorporates Arabic-specific features in the recognition algorithms which can be leveraged for handwritten Arabic recognition. As a consequence of these early design choices, it is plausible that ACRS can provide a solid foundation for the development of a handwritten recognition system. If true, the development time of such a system can be minimized – a substantial benefit in terms of availability and cost. However, at the outset, it is not certain that ACRS can be leveraged in this way, so our development work begins with a feasibility study

whose principal goal is to validate this approach and to provide insight into the development tasks required to produce a high-performance handwriting recognition system, using ACRS technology as a starting point.

This paper thus describes results obtained as part of a handwritten Arabic OCR feasibility assessment. The study was designed to gauge the performance of ACRS when it is used for Arabic handwriting recognition. In the following, the acronym HACRS refers to ACRS when it is used for handwriting recognition. HACRS 1.0 is actually identical to ACRS version 9.1, however, because no modifications were made to the machine-print system before obtaining the baseline measurements for handwritten input.

The technical assessment of the study consists of two components: (1) character and word accuracy of non-degraded (“clean”) document page images, and (2) character and word accuracy of degraded images. For each component, the origin of specific OCR failures have been identified and used to form a realistic development plan. This report primarily addresses the second component concerned with degraded pages since its results are more indicative for real-world applications.

Section 1.1 highlights some issues concerning Arabic handwriting recognition while Section 1.2 outlines the architecture of ACRS and its individual modules. Section 2 summarizes the feasibility study, and Section 3 highlights recommended tasks for the development of ACRS as a handwriting recognition system. Section 4 briefly describes the initial performance gains obtained primarily from training the recognition system on handwritten Arabic.

1.1 Handwritten Arabic Recognition

Handwritten Arabic documents present special challenges for character recognition. Like handwriting in other languages, many Arabic handwritten texts are unreadable or barely readable by native readers of the language – often context is needed for correct interpretation. The example shown in Figure 1 illustrates some issues that arise in handwritten Arabic.

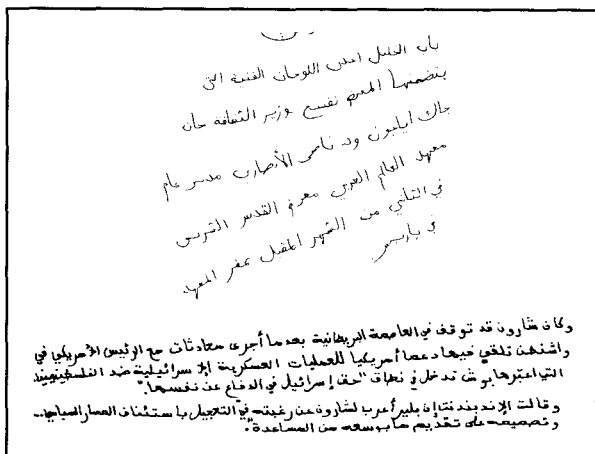


Figure 1. Two examples of Arabic handwriting

First, location and normalization of text lines is much more complicated in handwritten Arabic than machine-printed Arabic. Within a single machine-printed document zone, line spacing is even and predictable. Machine-printed lines are normally straight, except for skew introduced during scanning. In contrast, handwritten documents may contain tilted, uneven, and overlapping lines. Flourishes and other non-text marks may connect one line to another, or introduce extraneous lines. Uneven writing introduces additional ambiguities in line position and extent.

A second major complication is the variation in character forms appearing in handwritten Arabic. Although there are many machine-printed Arabic fonts, each is reasonably predictable and machine-printed documents are fairly consistent in the character forms they contain. Handwritten documents reflect the eccentricities of individual writers, as well as regional differences. As in other languages, letters may be obscured or left out in documents written by hand. All of these factors increase recognition difficulty.

1.2 ACRS Background

Although OCR technology has been under development and successfully commercialized for decades, significant limitations still exist in the application of OCR to real-world document collections. Scanned or faxed documents comprise important sources of information for government applications. The resulting images are often noisy, low in resolution, and of poor image quality, due either to the quality of the paper originals or to artifacts introduced by imaging and transmission. Unfortunately, most commercial OCR systems are designed to read high-resolution, high-quality document images. Generally, such systems perform poorly on faxed or low-resolution scanned documents. Additionally, "off-line" OCR systems ordinarily are not designed to handle the connected characters used in scripts such as machine-printed Arabic or handwritten Arabic and English.

A fundamental limitation to recognition performance in most OCR systems is a high character segmentation error rate. In low-resolution scanned or faxed documents, recognition errors are introduced because character separations are not resolved by the imaging device or characters are connected by image artifacts. Character segmentation failures represent the major source of misrecognized characters in low-quality documents. In such cases, the performance of the OCR system often degrades to the point where the output becomes unintelligible. These problems are even more significant for OCR on script documents, where there may be no spaces to separate characters.

Typical OCR systems produce a pure text output data stream, for example ASCII or Unicode. A single top-choice result is produced for each character, regardless of the confidence of the recognition. In reality, even though the top choice may be incorrect because of image quality or artifacts, a second choice may be correct but it is not included in the output. Comparison against a lexicon of possible words may allow recovery from top-choice errors, but only if the lexicon contains the word to be recognized. Proper nouns and technical jargon are rarely included in general-purpose lexicons. As a result, OCR errors are more probable on specialized query terms of most significance to document retrieval applications.

ACRS has been designed from the outset to overcome the limitations of standard OCR systems. In fact, ACRS accommodates low-resolution, low-quality text through special page image processing algorithms. The recognition processing modules have been developed specifically for script languages in order to minimize segmentation errors. Internally, the system generates multiple recognition hypotheses in order to maximize the probability of correct recognition when a lexicon is available or when application logic provides a basis for choosing among multiple character hypotheses.

The ACRS processing flow is shown in Figure 2. At a high level, the system consists of two stages: Page Processing and Recognition Processing. Page processing operates on the input image to verify the language, clean up the image and remove artifacts, break down the image into regions or zones of text, and finally extract lines of text which are processed for recognition. Each module has been tuned to detect and reject image artifacts and noise. The ACRS modules are described further in the following subsections, with an eye to Arabic handwriting recognition.

1.2.2 Image Preprocessing, Page Decomposition, and Line Segmentation

During *image preprocessing*, a TIFF-formatted binary image of a single page or zone is transformed into an efficient internal representation for high-speed processing with a low memory footprint. The image is first analyzed to determine whether it is a fax. If a fax

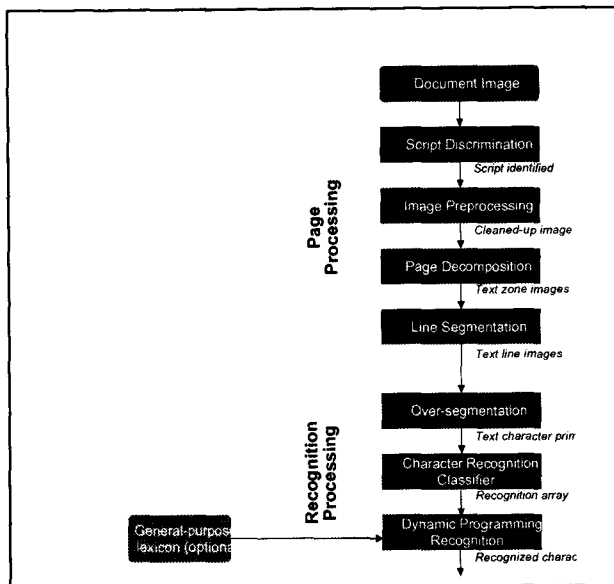


Figure 2. ACRS approach to recognition of individual lines.

header is detected, it is located and separated from the image with the header text contents possibly added to a metadata file. The noise content of the image is measured and a decision model is used to determine whether noise filtering is necessary.

If required, a *noise removal* module is applied; this module is aggressive in removing even heavy salt-and-pepper noise, without removing content that may be dots associated with Arabic characters. Once the image has been cleaned, overall page characteristics such as gross orientation and fax type are assessed. Orientation and the aspect ratio of low-resolution faxes are corrected, if necessary. Finally, small angle image skew (rotation) is measured and corrected.

Page decomposition is then performed on the de-skewed image to segment the page into zones with similar characteristics such as size and stroke width. Graphics, images and other non-text items are removed from further recognition processing. Each text block is then partitioned into individual lines of text by the *line segmentation* module. The text line images are normalized and passed to Recognition Processing.

1.2.3 Over-Segmentation

An *over-segmentation* recognition strategy has been implemented to overcome the limitations of traditional character segmentation when presented with low-quality imagery or connected script. The goal of over-segmentation is to split, or segment, an image of text into *primitives*, pieces containing an individual character or a portion of a character. The task of correctly assembling primitives into characters is performed later in conjunction with character recognition. A primitive that contains multiple characters represents a segmentation error because correct individual characters cannot be reconstructed from it. Therefore, the over-

segmentation algorithm is tuned to produce character fragments, rather than multiple characters, by incorporating language-specific characteristics such as the strong baseline in Arabic text.

The primitives are reassembled into various combinations, called *unions*. Unions are formed from con-

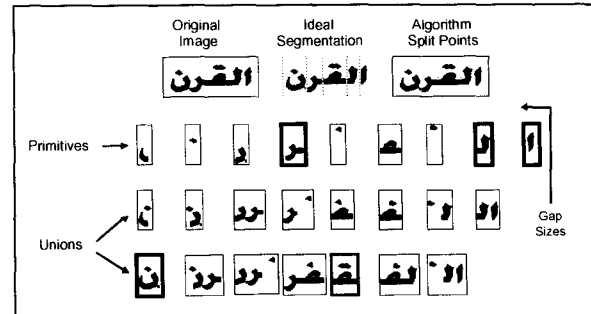


Figure 3. Over-segmentation and the formation of multiple segmentation hypotheses. Heavy box borders indicate correct characters.

tiguous primitives, preserving their spatial relationships. Among these unions are the ideally-segmented characters, as illustrated in Figure 3. The union images are passed to the character recognition module, which simultaneously determines the final segmentation and recognition outputs.

1.2.4 Character Recognition

During character recognition, each union is processed by a *feature extractor* to obtain a vector of descriptive attributes. The feature extractor is a module designed to systematically extract particular characteristics (features) that discriminate one character from another in a given script, for example, character curvature or the presence of dots. These features are then passed to a classifier trained on features from characters of the appropriate language and script. Obviously, a key question with respect to handwriting recognition is whether the features used for machine-print Arabic are fully effective or whether a more suitable feature set must be found.

The output of the classifier is the set of probabilities linking each union to each of the valid characters. The use of probabilities can dramatically reduce the breadth of search in the subsequent dynamic programming step, because the majority of entries are zero and can be ignored. The classifier probabilities for each of the unions are stored in a 3-dimensional array, termed the *recognition array*.

1.2.5 Dynamic Programming

The recognition array is passed to a *dynamic programming* algorithm, which transforms the information in the recognition array into a sequence of characters,

performing both final segmentation and recognition. The dynamic programming algorithm determines the sequence of unions that yields the highest cumulative character confidences across a line. One benefit of the dynamic programming framework is that the character confidences supplied by the classifier may be augmented with bonuses or penalties based on supplementary information such as a general-purpose lexicon, image-domain information such as character spacing, or linguistic constraints such as the rules for assigning language-specific character forms.

2 Feasibility Study

As noted earlier, this paper focuses upon results obtained as part of a handwritten Arabic feasibility study. The following sections summarize what was observed when processing degraded pages of handwritten Arabic.

2.1 Experiment Design

In the degraded page study, the effects of degradation on recognition performance, and the differences between performance results for individual writers were examined, both on the clean "Accepted" set images, and on the mixed "Degraded" set.

For the Degradation Tests, character and word recognition performance were measured for the 496 images in this data set, on the 5 different treatment categories: Clean(none), Skewed, Streaked, Speckled, and Darkened—and then characteristics of the degraded images were analyzed to determine what conditions, in interacting with the ACRS recognition software, might explain the performance differences found.

For the Writer-Specific Analyses, results for character and word recognition rates per page, for each writer, were examined for both the Accepted and the Degraded image sets. The correlations of the character and word rates in terms of the rankings for each writer were also considered. Following this, the writing of those writers representing extreme cases in the distributions were reviewed to look for characteristics that might explain why these were outliers.

2.2 Degradation Tests

Results of the performance analysis for the five different degradation treatments, both character and word recognition rates, are shown in the attached statistical tables and histogram plots found in Appendix A ("Performance Results Charts"). These are placed in order of our expectations for performance (best to worst), namely: Clean, Skewed, Streaked, Speckled, and Darkened. The plots and tables both show the numbers or percentages of pages having particular word or character recognition rates. Cumulative and inverse cumulative plots are also included to make it easier to find the medians, and the numbers of pages having scores more or less than a given percentage.

Overall, these results on degradation are quite good, even somewhat better than expected. For all of the degradation modes except for the worst (Darkened), the decrement in performance is fairly minor, and for the Darkened case, the character recognition rate drops by only a fourth (mean) or fifth (median), while the word recognition rate drops by about half. This is significant because many systems would lose much or nearly all of the text in the Darkened case. Moreover, for the "Speckled" case, performance actually *improved* slightly, over the Clean case. These two results, in particular, are a direct result of the noise analysis and removal capabilities present within ACRS. And, it is worth emphasizing that these results are for handwritten data, using a system that has been designed and trained to operate on *machine-printed* text.

2.3 Experiment Interpretation

A more detailed analysis of these degradation test results reveals some interesting aspects. On the character recognition side, compared to the Clean case (Median 32.9, Mean 35.8, Max 66.9), the Skewed case is very close (Median 31.5, Mean 33.3, Max 61.43). In fact, our expectation is that normally the Skewed images would show *no* performance differences at all, *except* for the fact that some of the text is cut off by clipping, and is thus completely lost. The figures we see here may therefore represent close to an upper bound on performance for the Skewed set, in terms of what text is actually left on the page. In other words, these small differences may be almost entirely due to outright loss of text, rather than difficulties in handling skewed pages.

For the next, Streaked case (Median 29.4, Mean 31.3, Max 63.9), there is a drop of a few percentage points—again fairly minor. The additional errors fall into two groups. For the horizontal lines, even though algorithms to ignore underlines are implemented in ACRS, if the line is connected to a descender, there may be cases where it would not be correctly eliminated, and this would interfere with the recognition of that word and its letters. For the vertical lines, there is a detection and removal algorithm for purely vertical lines such as those that might be generated by a fax machine. This should do a good job of removing most of vertical lines, however, in some cases, these lines are not perfectly vertical and thus might not be adequately detected and removed by the current algorithm.

The Speckled case (Median 34.7, Mean 35.5, Max 70.3) provided a real surprise. While performance was expected to be slightly worse than the above two, in actuality it was slightly *better* than the Clean result, by a couple percent, at both the low and high end. A careful examination of these cases revealed two things. First, the degree of noise is generally light, and the noise is generally less dark than the text—making it fairly easy to remove with our binarization algorithms.

In addition, something not noticed earlier is that the method used to create the speckle also apparently thickens the strokes of the characters somewhat, making them slightly easier to recognize, as well as joining the characters of words together more tightly. The latter effects apparently led to the observed improvements.

Finally, for the Darkened case (Median 25.6, Mean 26.8, Max 66.9), there is a larger drop in performance, as one would expect given the large amount of dark noise that is added to the images—about 7 percentage points in the Median (20% lower), and 9 percentage points in the Mean (25% lower). Still, these are not very significant drops considering the extent to which the quality of the image has deteriorated. Again, this ability to recover text in the face of fairly heavy noise is due to the noise assessment and filtering algorithms present in the ACRS system.

On the word recognition side, the picture is somewhat the same. For the Clean case, the performance on the 496 image set used for this test is reasonably good for handwriting, using a machine print-based system (Median 9.1, Mean 11.9, Max 42.4). The Skewed case is barely different (Median 9.1, Mean 11.5, Max 36.2)—the main differences are felt on the high end (Max 36.2 compared to 42.4), as one might expect due to the clipping effects cited earlier. The Streaked case shows more degradation in performance (Median 6.9, Mean 9.5, Max 40.8), but only a couple of percentage points, again due to the errors in detecting and removing the spurious lines that were mentioned above.

As with character recognition, word recognition rates on the Speckled case (Median 10.7, Mean 13.3, Max 56.1), were actually a bit better than even the Clean case, especially for the best pages. Again, it appears this was due to the thicker characters, leading to better intra-word joining, that resulted from the speckling treatment of the page. Finally, the Darkened case showed a more significant decline, as expected (Median 3.9, Mean 6.8, Max 46.2) for the averages—however, these represent a drop of only 40-60 percent in performance levels—not bad considering the severity of the degradation, and, on the best pages, some *improvement* over the Clean case was found, again likely due to the thickening of the character strokes, similar to the Speckled case.

2.4 Writer-Specific Analyses

The second set of analyses had to do with looking at performance differences between individual writers. These results are summarized in charts which also appear in Appendix A. The first pair of these shows the writer IDs sorted by their Accepted set character recognition performance along the x-axis. We then have the character recognition rates for the Accepted and Degraded image sets in one graph, and the corresponding word recognition rates for the two image sets in the other graph.

These charts show that the deterioration in character and word recognition performance for different writers, actually varies to some extent. Some (such as DR, AA, MC, and AH) are hit harder than others by degradation, while a few (MS, JB, and JC) actually seem to get better. (Though the fact that these are at the low end to start with suggests that the improvement seen may be due to the fact that their worst pages were thrown out in going from the 568 Accepted image set, to the 496 Degraded set). As it turns out, AA and MC degrade more rapidly because their writing is small. DR has taller lines, but this comes mainly from large ascenders—the interior characters are also small and thus easily damaged by degradation or noise. AH suffers under degradation for a different reason—the text lines in this writing are sloppy, not straight, and not well-separated, so that when noise is added, it becomes very difficult to segment them from each other.

It can also be seen from the word recognition chart that character and word recognition performances do not go strictly hand-in-hand.

In the second pair of charts, the correlations of the character and word recognition *rankings* of the 19 different writers are shown, in one case for the Accepted image set, and in the other for the Degraded image set. What the charts show, in particular, is where the two recognition rates are distinctly *different* for certain writers, based on specific characteristics of their writing. The fact that Accepted results differ from the Degraded results, indicates that different writing styles have varying abilities to survive the effects of degradation. Thus, in attempt to summarize the above discussion about performance rates for different writers, it appears there are a small number of basic principles which explain most of the differences seen over all four charts, some of which are more obvious than others. They are:

- *Well-formed, distinct characters will be more easily recognized than sloppy ones, and better able to survive degradation*
- *Likewise, well-formed, distinct words will be more easily recognized than sloppy ones, and more robust with respect to noise*
- *Small letters, if written in a very regular fashion, look more like machine-print text and are likely to be better-recognized by our machine-print text recognizers*
- *Small letters, because of their size however, are less able to survive degradation, especially severe degradation*
- *Tall letters, if written well with uniform scaling, can also be fairly easily recognized, but can also survive degradation processes intact enough to still be recognized*
- *Some writers have very tall ascenders and descenders, but with other characters being*

fairly small. This can reduce character recognition rates on the smaller characters, because after normalizing a line or word to the larger character size, the smaller ones are too small to be easily recognized.

- *On the other hand, some writers who have this style of writing (which is often also very script-like, sometimes with "elided" characters that are barely present—almost like cartoon strokes)—some of these writers also naturally emphasize "key" letters that are needed to recognize the word. If there is enough information in these emphasized letters to clearly identify a word in the language, then the word may still be recognizable, even if some of the characters are not.*
- *Some writers make the separation between words very distinct, and/or tie together the letters of the same word, either in terms of proximity or actual connection. This improves word recognition.*
- *Other writers write the characters of a word in a more spread-out or broken fashion, and often have breaks between the words that are not obviously larger—this makes it difficult for the character segmenter to assign word breaks accurately, especially after degradation.*
- *Some writers make straighter lines of text, while others make curved or skewed lines; the latter are more difficult because the line segmenter is currently built to handle straight, uniformly oriented, machine-print lines.*
- *Some writers put more separation between lines than others. When there is less separation, especially in cases of strong degradation in the form of noise, it is more likely that line boundaries become confused, and the lines possibly merged, which basically loses all of the text in both.*

In looking at the rank correlation charts, it can be seen that DD, JP, and JPH on the 568 Accepted set chart have better word recognition rates than their character rates would suggest, while PP and JM are worse. DD and JP are examples of very script-like writing, with elided characters, but such that key characters are emphasized, allowing word recognition to occur, despite poor character rates. JPH appears to do better, because the word separations are particularly distinct in this writing, while the characters are only moderately good in terms of recognition. For PP and JM, on the other hand, while the characters are reasonably well-formed, the words are somewhat sloppy, and not well-separated.

For the 496 Mixed Degraded image set, on the other

hand, we see in the second chart of this pair that JPH and AH perform better than expected in terms of word recognition, while BW and JM perform worse. Again, JPH, as mentioned earlier, has very good word separation, which augers well in the face of noise and other forms of degradation. AH was lower in both character and word performance under degradation, apparently due to line segmentation issues but, in relative terms, the word rate here survived better because the letters are large and well-written, and the words are reasonably well-separated, even though the lines are somewhat sloppy and not well-separated. BW does more poorly because the writing is small, and there isn't good separation between the words—which gets worse when noise is added to the image. Similarly, JM, as noted earlier, has good characters but the words are sloppy, somewhat broken, and not well-separated, and the lines are not well-separated—all of which lead to segmentation problems when noise is added.

3 Recommended System Development

The ACRS feasibility study results described in Section 2 show that it is very likely to be feasible to construct an effective Arabic handwritten recognition system, using ACRS as the foundation. In fact, the study provides good reason to believe that well-understood engineering modifications to ACRS will result in a high-performance system that is capable of processing a wide range of real-world handwritten material. The results of the feasibility study also highlight the processes within ACRS that require adaptation, extension or replacement before handwriting can be recognized reliably. Most important among these processes are line segmentation, character segmentation and recognition, and word recognition.

Based on the encouraging results of the feasibility study, the envisioned handwriting recognition system will be developed on the foundation of the existing Arabic Character Recognition System. The page and recognition processing capabilities of that system will be extended to accommodate the unique characteristics of *handwritten Arabic*.

More specifically, the adaptations of the ACRS system needed to account for handwritten material are summarized in Table 1.

Module	Requires Modification	Comments
Script Discrimination	Yes	Tune existing handwritten vs. machine-print discriminator
Image Preprocessing	Yes	Tailor page 'characterization' to handwritten Arabic
Page Decomposition	No	Use existing ACRS capability
Line Segmentation	Yes	Modify for handwritten lines
Over-segmentation	Yes	Modify for handwritten segments
Character Recognition Classifier	Yes	Retrain character classifier Reevaluate ligature handling
Dynamic Programming Recognition	Yes	Tune for modified classifier outputs

Table 1. Modifications to the ACRS for Handwritten Arabic Recognition.

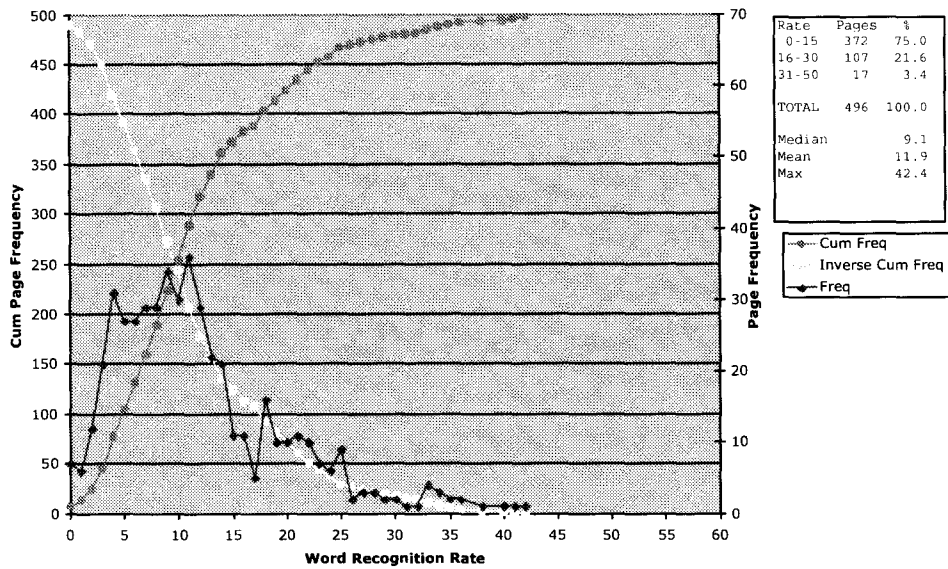
4 Current Status of Handwritten Arabic

This paper has focused on the results of a handwritten Arabic feasibility study carried out by NovoDynamics earlier this year. Since the completion of the study, the development of a handwritten Arabic system based on ACRS has begun, focusing on the modifications noted in the previous section. To date, the results have been excellent. Specifically, the current character recognition rate is approximately 55% (i.e., a 66% increase) even though the newly developed handwritten training and test materials are much more challenging with respect to writing style and neatness. These developments will be the subject of a future report.

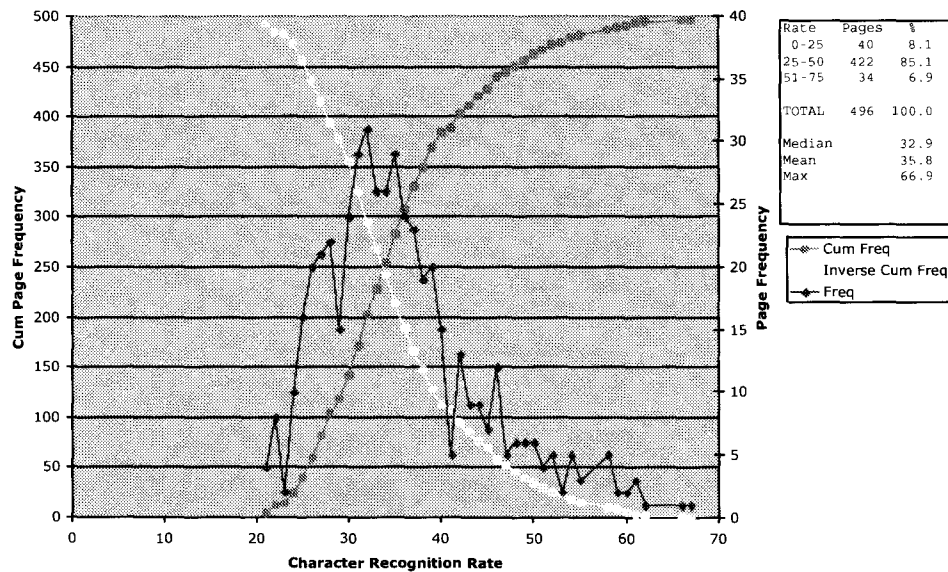
Appendix A: Quantitative Results

A.1 Degradation Test—Clean Set

Word Rate on Clean Images

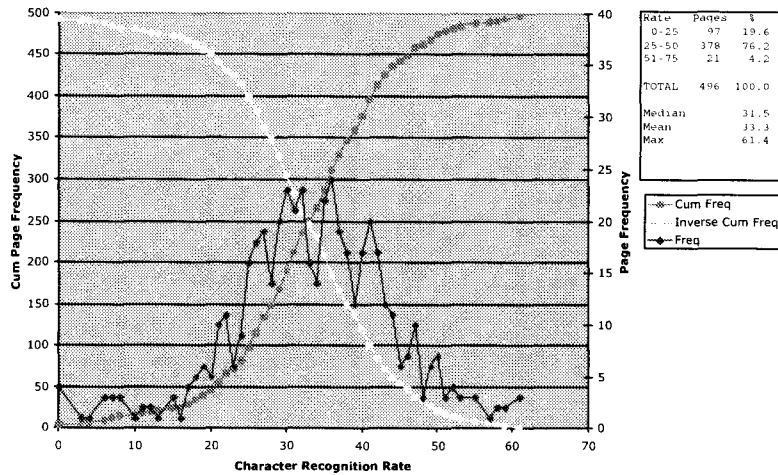


Character Rate on Clean Images

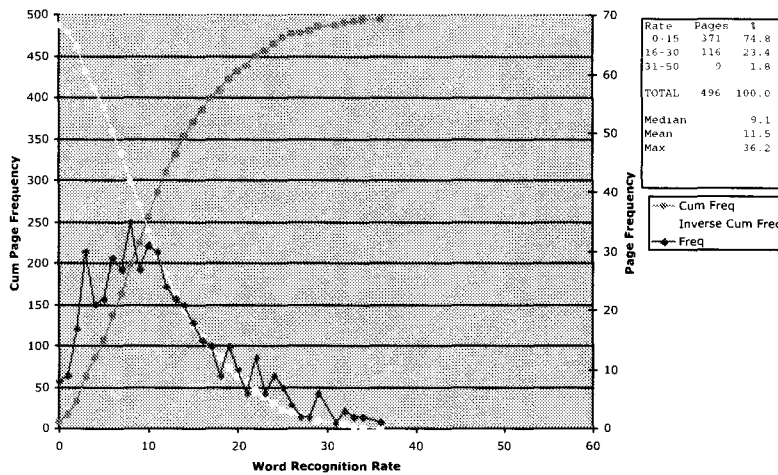


A.2 Degradation Test—Skewed Set

Character Rate on Skewed Images

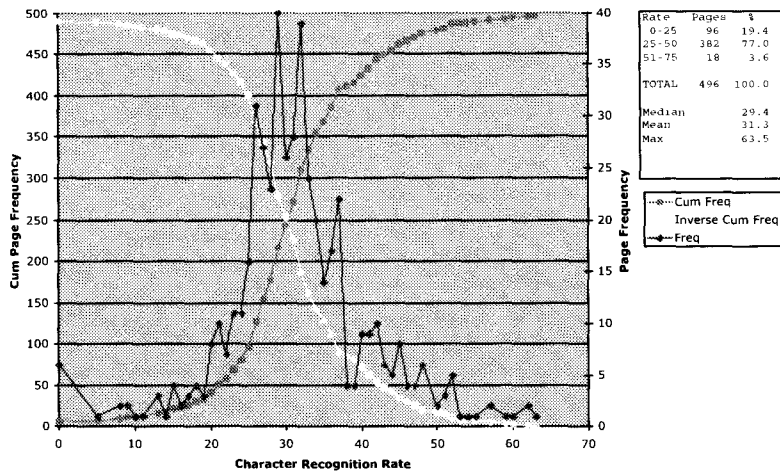


Word Rate on Skewed Images

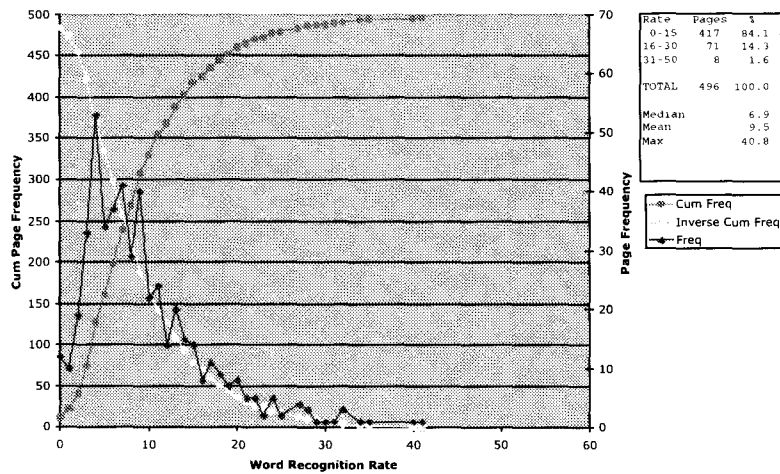


A.3 Degradation Test—Streaked Set

Character Rate on Streaked Images

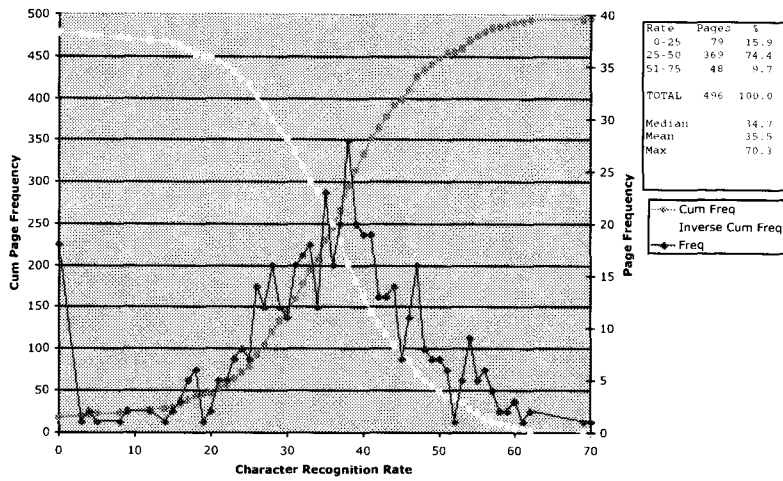


Word Rate on Streaked Images

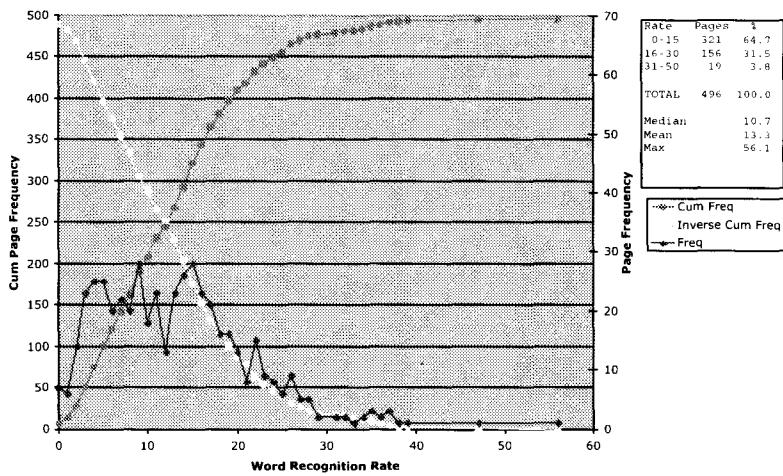


A.4 Degradation Test—Speckled Set

Character Rate on Speckled Images

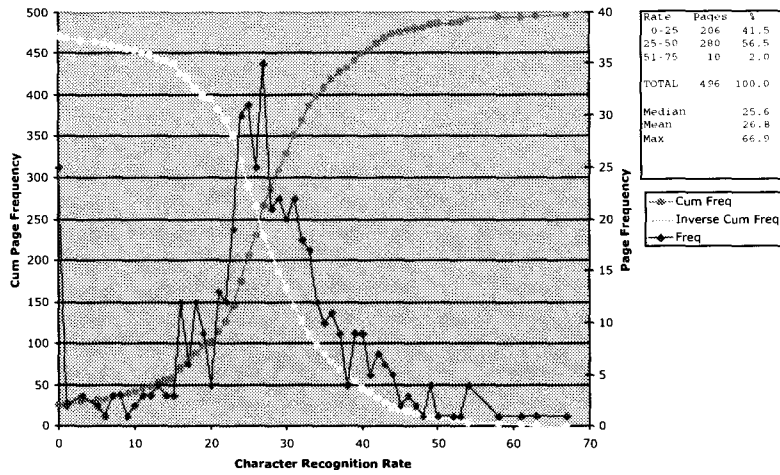


Word Rate on Speckled Images

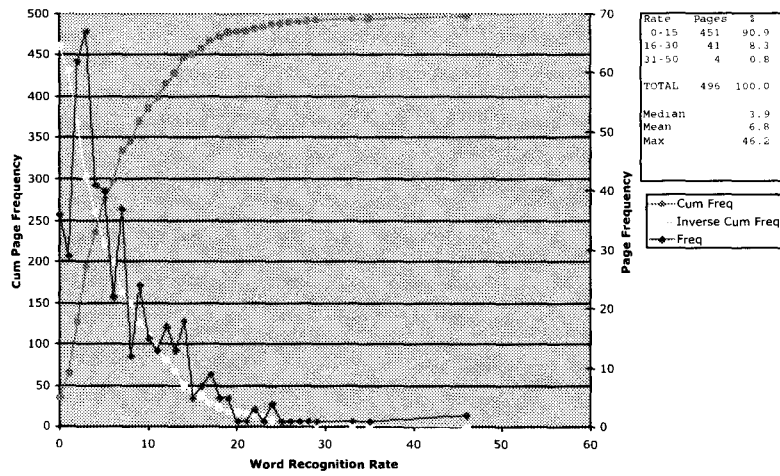


A.5 Degradation Test—Darkened Set

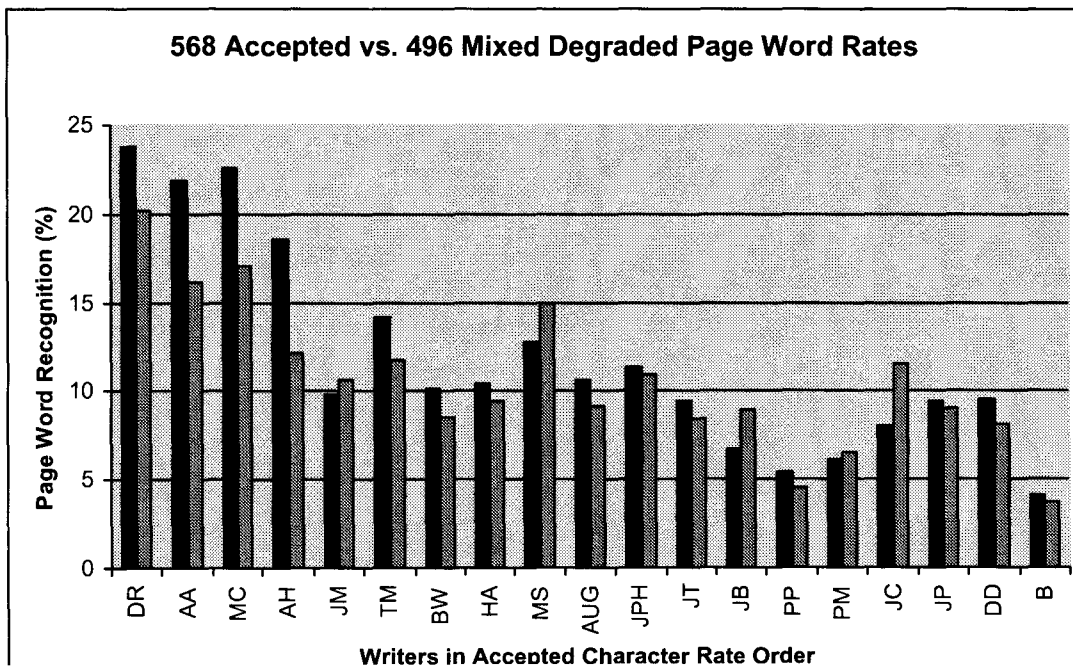
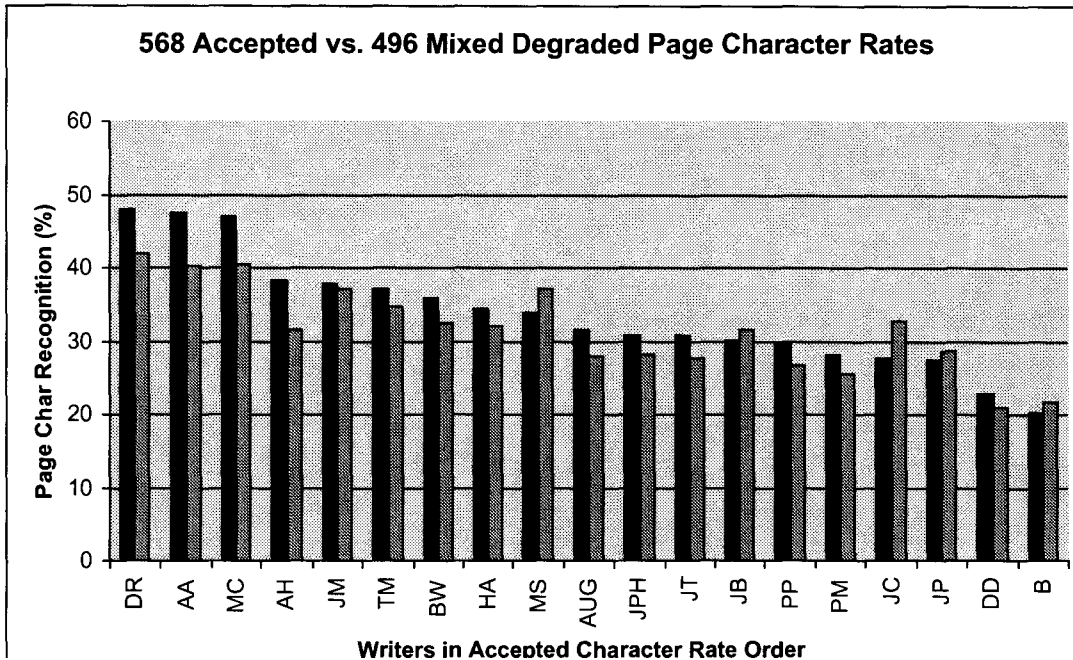
Character Rate on Darkened Images



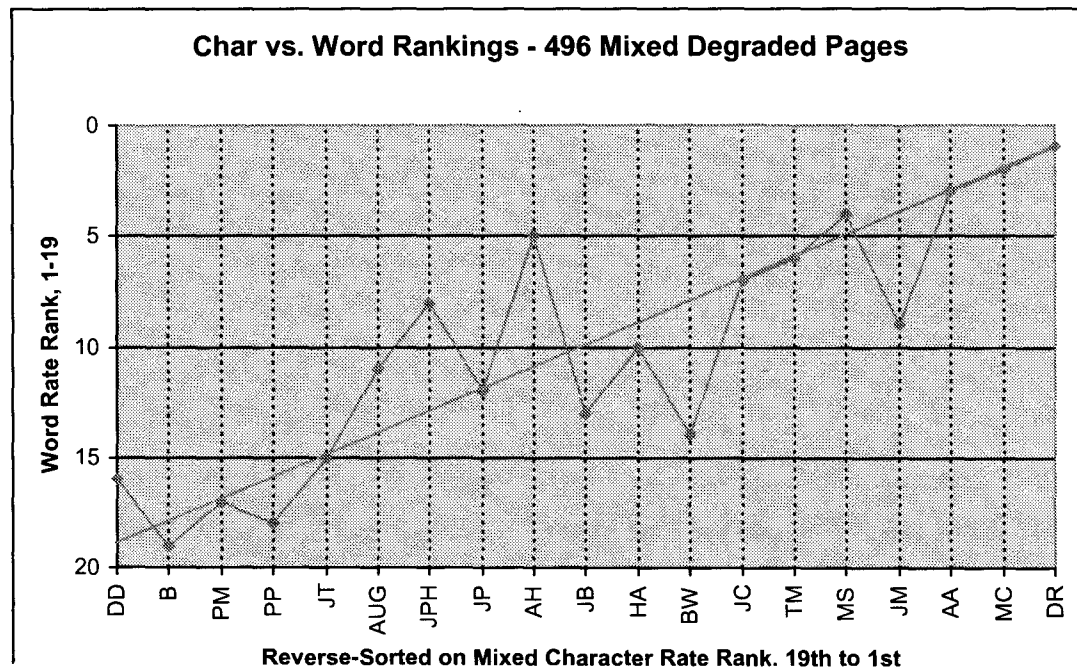
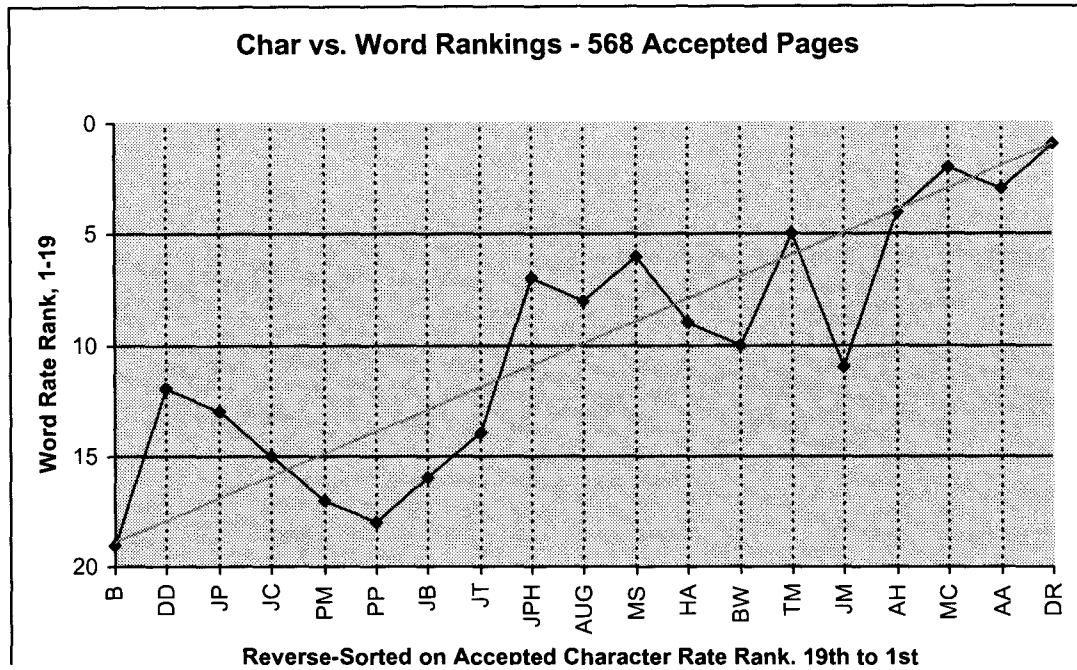
Word Rate on Darkened Images



A.6 Writer Assessment—Performance Rates



A.7 Writer Assessment—Rank Correlations



Cross Domain Applications

Robot Navigation Techniques for Engineering Drawing Analysis

Thomas C. Henderson Chimiao Xu
School of Computing
University of Utah
Salt Lake City, UT 84112 USA
tch@cs.utah.edu

Abstract

Engineering drawings provide a significant challenge to image analysis. The goal is to take a scanned engineering drawing image and produce an interpretation of the contents in terms of characters, digits, arrows, line segments, dimensions, etc. Our goal is to incorporate these results into a legacy system re-engineering process (e.g., image analysis must provide parameter values for manufacturing features like counterbore holes, etc.). We propose to treat the problem in two steps: (1) determine a good set of feasible points which constitute the markings on the page, and (2) subsequently treat these as a 2D floor plan that is explored by a tiny mobile agent that navigates through the drawing and produces a map. The trajectories followed by the agents allow a segmentation of the image into basic geometric units: straight line segments, end points, branch points, etc. In this paper we study static Pseudo-Range Maps (PRMs) to identify point features in the image.

We believe that this approach can be applied to any kind of line drawing material where the objects of interest can be segmented. We conjecture that it will also be possible to characterize and identify processes that create such line drawings (e.g., people, printers, etc.).

1 Introduction

Almost all legacy engineering designs involve 2D CAD drawings. The most common tasks for the re-engineering of legacy systems include the modification of existing designs to make a new design, or the synthesis of 3D models from multiple 2D views (usually available as engineering drawings). Automatic engineering drawing analysis offers a useful tool in the interpretation of engineering drawings when the original electronic CAD is not available. Engineers engaged in reverse engineering rely heavily on the scanned engineering drawings from blueprints as well as 3D data from the available

physical parts. Our goal is to aid the design process by achieving the most accurate image segmentation possible in order to permit the best interpretation of annotations in digitized images of CAD drawings.

The automatic analysis of engineering drawings has posed interesting and challenging research topics in document analysis, pattern recognition, image processing and computer vision for a few decades [19, 22–25]; however, 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.) Most previous work deals with the interpretation of particular symbols and structures in CAD drawings (e.g., straight line vectorization [4, 12, 16], arcs [20], and dimensioning analysis [3, 5]). Our survey shows that previous research focuses on methodologies and algorithms for individual aspects of the problem and only achieves good overall performance in special circumstances. Most of the known algorithms and procedures require noise-free conditions which is usually an unrealistic assumption. Research is still underway in this area due to real world application demand, and no existing system works well universally.

We have studied this problem for a number of years (e.g., see [8–11, 22]). That work explored the use of structural constraints in the underlying document elements, and feedback mechanisms to reinforce paths to successful interpretations. However, one constant issue is the poor quality of the image segmentation of the engineering drawings.

We propose to improve the image segmentation process by taking advantage of the fact that engineering drawings are human artifacts and thus have a regular structure. Moreover, this structure is quite similar in nature to the 2D floor plan of a building layout; e.g., lines are like hallways. Figure 1 shows a typical layout to be explored and mapped by a mobile robot. Similarly, the scanned digital image of an

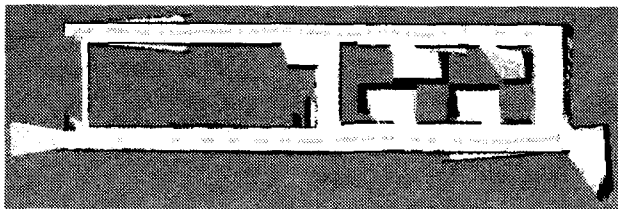


Figure 1: Building Layout for Mobile Robot Mapping

engineering drawing has errors and noise, and therefore poses difficulties for standard image processing techniques. We propose a paradigm shift: view the drawn lines and symbols as hallways, rooms, etc., and use robot navigation and mapping methods to analyze the drawing structure. Mobile agents are placed “inside” the drawn lines and located with subpixel precision. The gray levels serve as a *density* function, and a pseudo-range scan can be performed at any pixel (the background of the line drawing image serves as the solid surfaces). The engineering drawing analysis is now defined as a robot mapping problem. We propose to demonstrate that features important to the drawing analysis can be extracted using this approach. Once these features can be robustly extracted, we believe that it is possible to obtain the line segments and structures necessary for high-quality drawing interpretation. The method also makes it possible to learn interesting structure within the image automatically. Finally, localization methods can be used to match models of known structures (e.g., letters, digits, arrowheads, etc.).

In this paper we explore the static use of pseudo-range maps at individual pixel locations to classify 0-dimensional features (i.e., point features), including: (1) *end points*, (2) *interior corridor points*, (3) *corner points*, and (4) *branch points*. Performance of this approach is demonstrated on engineering drawing images for which ground truth is available.

2 Method

We propose a two-step process:

1. Threshold the image into foreground and background components, and
2. Use mobile robot mapping techniques to interpret point features in the foreground data.

Thus, step 1 produces a binary image and step 2 extracts point features of interest.

2.1 Foreground Segmentation

The images being analyzed are mostly comprised of lines; that is, elongated, thin linear structures. The foreground segmentation algorithm takes as input an image, I , and a length, k and is given as follows:

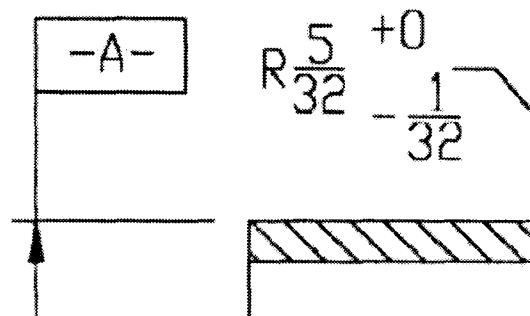


Figure 2: Original Gray Level Drawing Image.

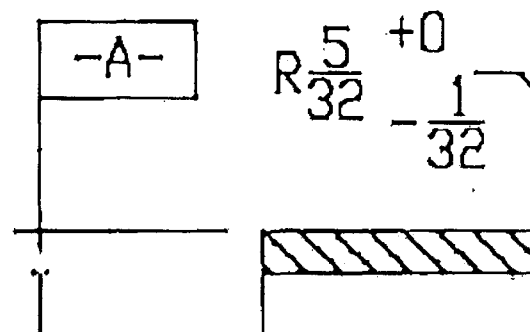


Figure 3: Segmented Image.

- for every pixel, p , in I
 - analyze the linear vector of k pixels centered at p and in the 4 major orientations (0, 45, 90, 135 degrees)
 - pick the orientation, if any, with the strongest contrast between the middle pixels and the outer pixels (outer pixels higher valued than the middle pixels since the drawing is dark on light background).

Figure 2 shows an original gray level image and Figure 3 its segmented counterpart. (Figure 4 shows a segmented image using a standard within-group variance thresholding technique [18].)

Note that the arrow head is missing in Figure 3 as it is not a linear structure. Also note how the fraction numbers and line are messy and not well-separated in Figure 4.

2.2 Mobile Robot Mapping Techniques

Figure 5 shows part of an engineering drawing with the position and orientation of a mobile agent overlaid on it. The agent can produce a pseudo-range map (PRM) at any location; Figure 6 shows the

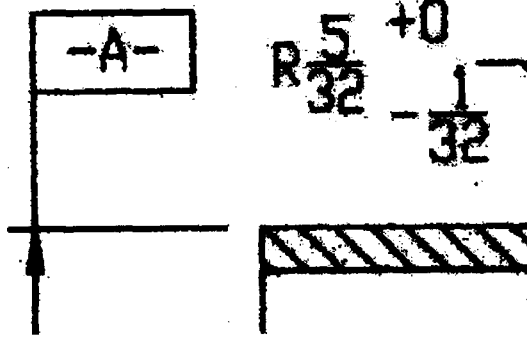


Figure 4: Image Segmented using Standard Method.

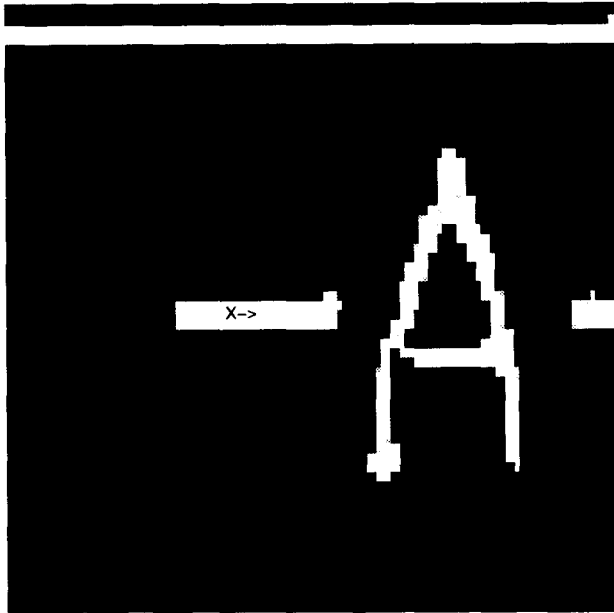


Figure 5: An Engineering Drawing with Agent Overlaid.

PRM for the given location (note that the range is taken at sub-pixel accuracy). The basic idea is that the agent will use the PRM in order to explore the line drawing. This will be done using standard mapping techniques. While exploring the line drawing, features, segments and symbols will be extracted. Performance will be analyzed using a dataset for which we have determined the ground truth.

Mobile robot navigation, localization and mapping is still an area of active research in the robotics community [2, 17]. However, many techniques are already developed which apply to our problem domain[7].

Mobile robot mapping techniques exploit three types of map concepts: *topological*, *geometrical*, and *grids*. From one or more of these standpoints, a map of the domain will be built. Specific approaches in-

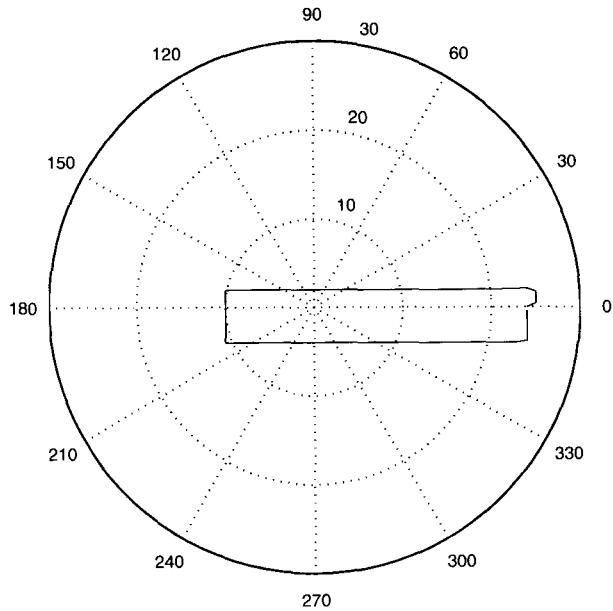


Figure 6: A Pseudo-Range Map (PRM) for the Agent Location.

clude the use of the visibility graph[14], the generalized Voronoi diagram (GVD)[1], trapezoidal decomposition, or probabilistic roadmaps[13]. We have developed some simple mapping procedures (similar to [7], p. 176) that construct an approximation to the GVD.

2.3 Feature Analysis

We describe here our work on 0-D features in the image, but ultimately, we hope to exploit navigation techniques to discover:

- 0-dimensional features: point or locale-centered features:
 - end points, small blob segments
 - corners
 - multi-branch points
- 1-dimensional linear features
 - straight line segments
 - curved line segments
- 2-dimensional features
 - arrowheads
 - certain symbols

An initial foray into 0-D feature extraction demonstrates the feasibility of this approach. The PRM can be used at each pixel to identify:

1. *endpoints*: the terminal part of a line segment (a dead end in terms of robot exploration)

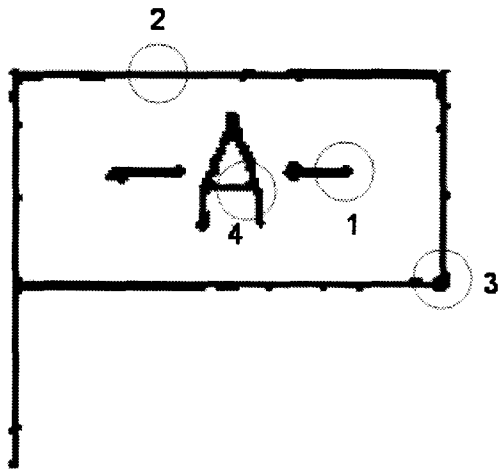


Figure 7: The four 0-D Feature Types (1: endpoint; 2: interior corridor; 3: corner; 4: multi-branch point).

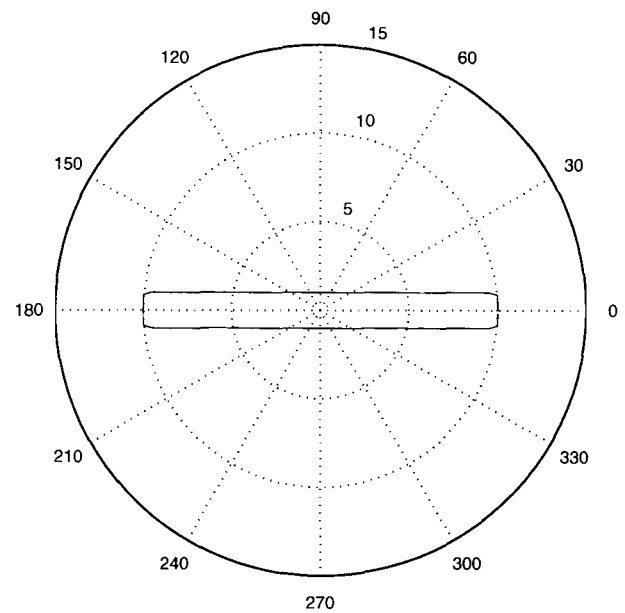


Figure 9: The Pseudo-Range Map for the Interior Corridor Feature (number 2 in Figure 7).

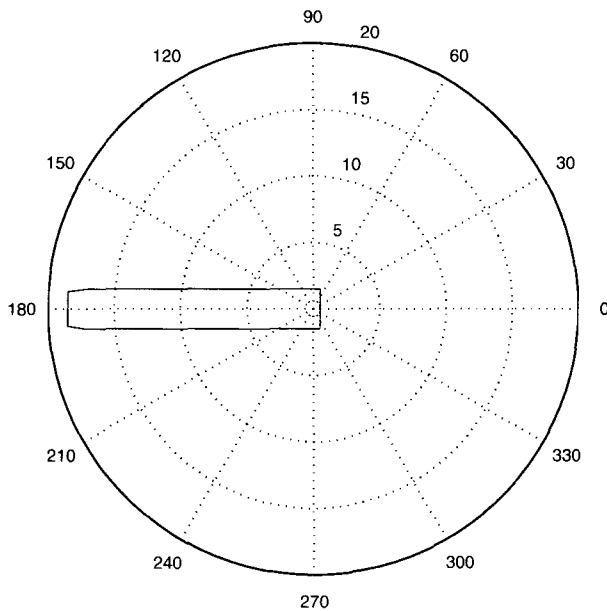


Figure 8: The Pseudo-Range Map for the Endpoint Feature (number 1 in Figure 7).

2. *interior corridor points*: two directions of travel possible, but not a corner (robot can move basically in two directions 180 degrees apart)
3. *corner points*: two directions of travel possible, but at significant angle off 180 degrees
4. *multi-branch points*: more than two directions possible for robot to explore.

Figure 7 shows examples of these four types, and Figures 8 to 11 show the PRM for each location circled in the image.

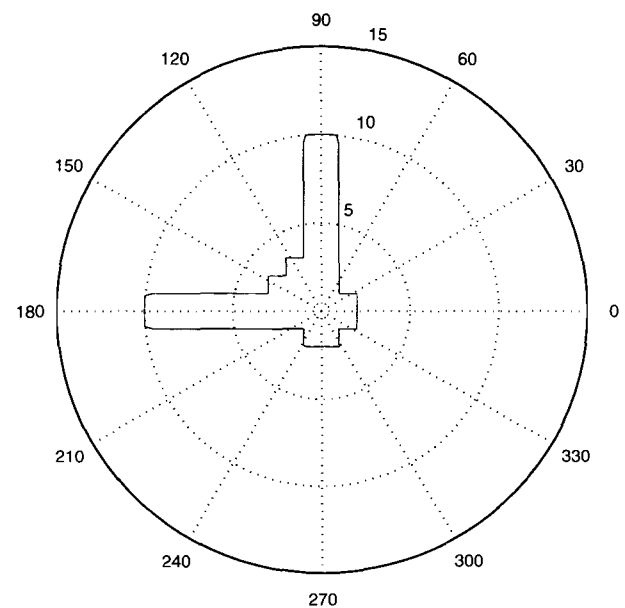


Figure 10: The Pseudo-Range Map for the Corner Feature (number 3 in Figure 7).

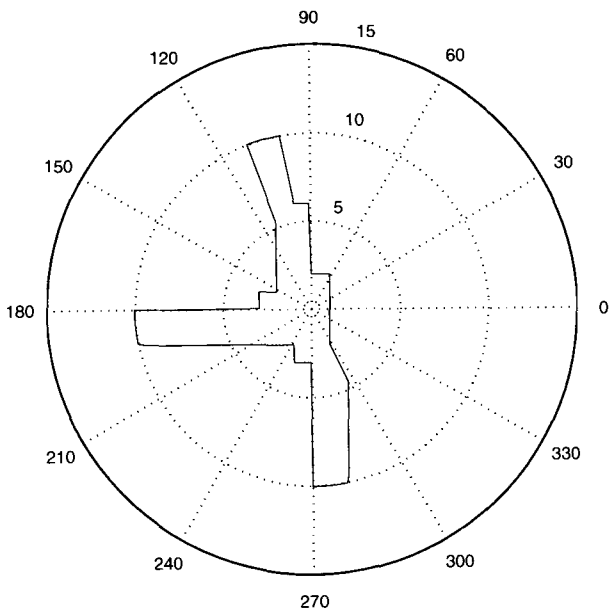


Figure 11: The Pseudo-Range Map for the Multi-Branch Feature (number 4 in Figure 7).

Given a thresholded image, this algorithm labels the 0-D features in the image; Figures 12 through 15 show the feature points extracted from the image. This is done using the polar range plots; the number of branches in the polar function tells the type of structure.

The feature classifiers work by taking into account properties of the PRM like number of peaks, maximum range, and range in the direction opposite to the maximum range (e.g., endpoint pixels have short range opposite the maximum, while corridor pixels have long range in both directions).

The performance of the proposed method, even without serious pre- and post processing, is better than that of the standard line detection algorithms.

In an attempt to determine how well our algorithm works, we explored the use of decision trees as a classification method. In particular, the information-theoretic approach described in [21] was applied, using the following attributes of the PRM:

- *attributes 1-7*: Hu invariant moments[15] of the PRM image region.
- *attribute 8*: area of the PRM image region.
- *attribute 9*: perimeter of the PRM image.
- *attribute 10*: sum of ranges in the PRM.
- *attribute 11*: total distance of the points in PRM perimeter to the point which is used to compute the PRM.

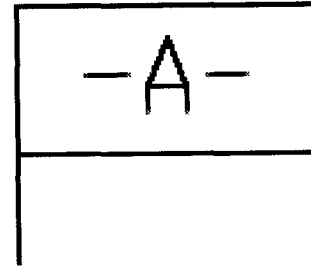


Figure 12: Endpoints Found in Image im00.

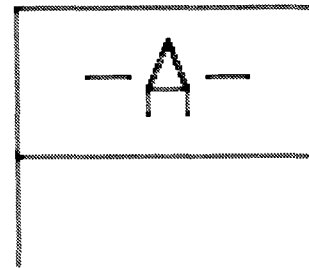


Figure 13: Corridor Points Found in Image im00.

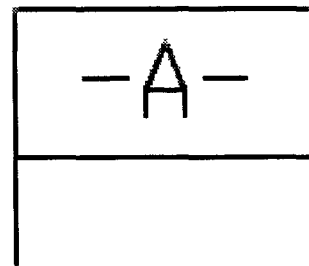


Figure 14: Corner Points found in Image im00.

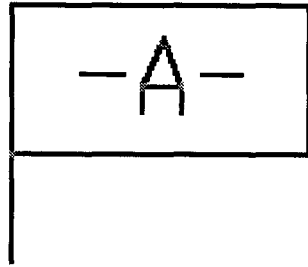


Figure 15: Branch Points Found in Image im00.

- *attribute 12*: sum of the absolute distance of the rows and cols of the PRM region point's to the row and col of the point which is used to compute the PRM, divided by the total number of points in the PRM region.
- *attribute 13*: number of branches in the PRM. (The range of the branch length must be longer than 90% of the maximum range of the PRM.)

A set of training samples were selected, and four decision trees were built: one for each 0-dimensional feature. It is interesting to note the most discriminating attribute (i.e., the first branch is determined by this attribute) for each feature: (1) endpoints: attribute 11, (2) corridors: attribute 2, (3) corners: attribute 12, and (4) branches: attribute 11. These classifiers perform well (see Section 3 on performance), but not as well as the hand written classifier.

3 Performance Evaluation

The overall system performance is evaluated in terms of the quality and computational complexity demonstrated over various image datasets. Noise is introduced during the digitization process; thus there are extraneous as well as missing objects in the resulting image. Both noise and missing data can have a large influence on the image interpretation process. In addition, blueprints might be stained, damaged during storage and usage, and scanners might have different blur, lighting, and scale factors [8]. To objectively measure how well the proposed system analyzes digitized engineering drawings, we compare results over a dataset for which we have ground truth knowledge acquired from engineering drawing datasets.

We have done the following steps for system performance evaluation:

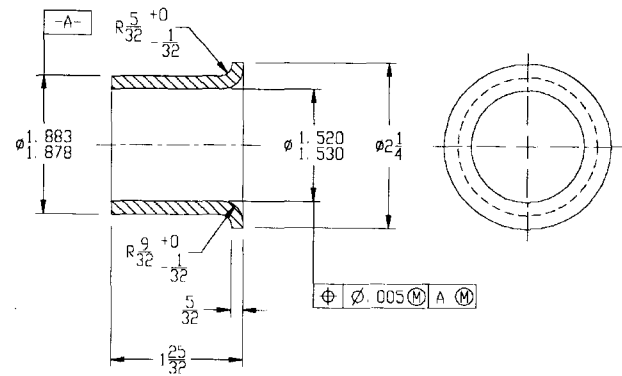


Figure 16: Test Image im00.

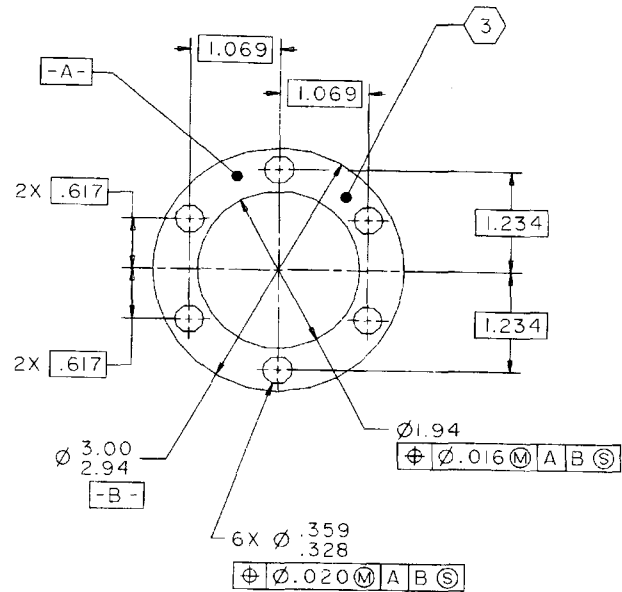


Figure 17: Test Image im17.

- Established a testbed benchmark set of five images and ground truth. These five images are shown in Figures 16 through 20.
- Run our previous feature classifier (non-PRM) on the test images. (See Table 1.)
- Run the PRM classifier on the test images. (See Table 2.)
- Developed a decision tree classifier using training data (numerical features of the PRMs). (See Table 3.)

Figures 16 through 20 show the engineering drawing test images.

To measure the performance of the algorithm, we use *recall* to mean the ratio of correct features found to total number of features, and *precision* to indicate the ratio of the number of correct feature responses to the total number of feature responses. Tables 1

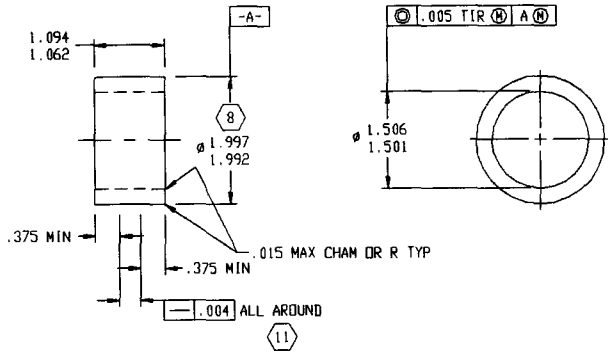


Figure 18: Test Image im18.

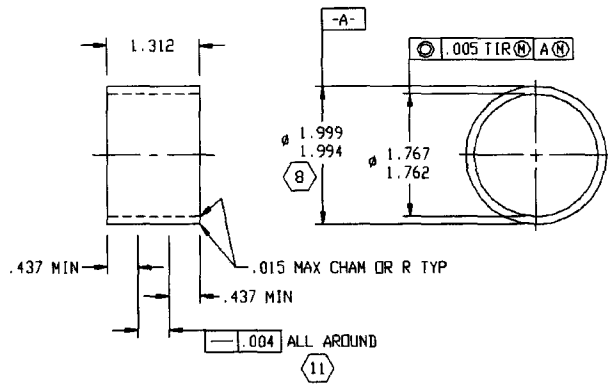


Figure 19: Test Image im19.

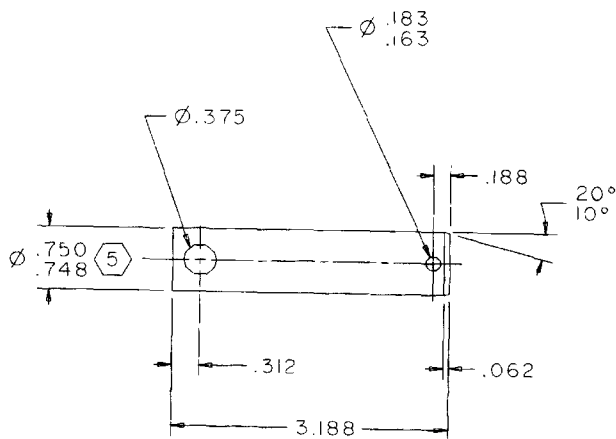


Figure 20: Test Image im22.

through 3 show the *recall* results from these experiments, while Tables 4 through 6 give the *precision* results.

Image	F_end	F_corr	F_corn	F_bra
im00	90.45	99.55	72.60	81.22
im17	92.41	99.25	65.11	89.25
im18	96.38	98.64	73.27	71.90
im19	95.59	98.95	73.88	86.36
im22	84.85	99.92	36.72	86.75

Table 1. Recall Percentage Correct for Non-PRM Method.

Image	F_end	F_corr	F_corn	F_bra
im00	100.0	98.70	100.0	100.0
im17	96.97	98.84	100.0	100.0
im18	100.0	99.07	99.63	100.0
im19	98.64	98.95	100.0	95.87
im22	97.32	98.09	100.0	98.92

Table 2. Recall Percentage Correct for PRM Method.

Image	F_endpt	F_corr	F_corn	F_bra
im00	100.0	87.61	85.48	99.53
im17	97.32	82.61	68.85	100.0
im18	99.55	84.91	84.49	80.99
im19	100.0	84.91	81.34	90.91
im22	97.73	91.85	46.88	97.59

Table 3. Recall Percentage Correct for Decision Tree Method.

Image	F_end	F_corr	F_corn	F_bra
im00	43.36	100.0	42.87	43.25
im17	35.10	100.0	28.64	41.52
im18	60.96	100.0	51.66	35.01
im19	53.94	100.0	63.23	34.92
im22	34.73	100.0	29.20	45.93

Table 4. Precision Percentage Correct for Non-PRM Method.

Image	F_end	F_corr	F_corn	F_bra
im00	35.82	100.0	42.59	50.11
im17	55.39	100.0	38.31	39.45
im18	69.89	100.0	47.83	57.56
im19	68.55	100.0	48.57	58.27
im22	31.06	100.0	28.20	44.50

Table 5. Precision Percentage Correct for PRM Method.

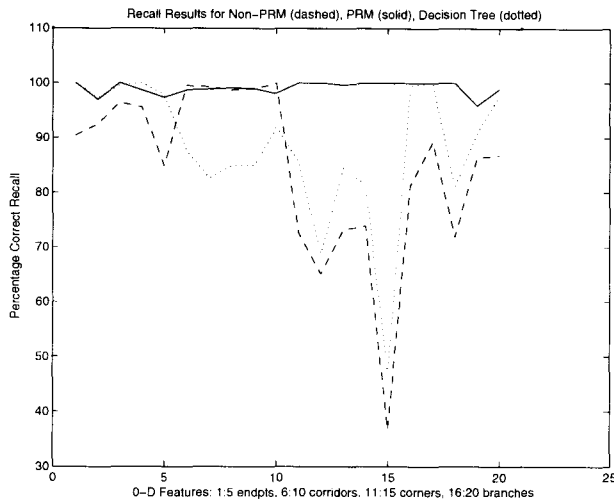


Figure 21: Percentage Correct Recall.

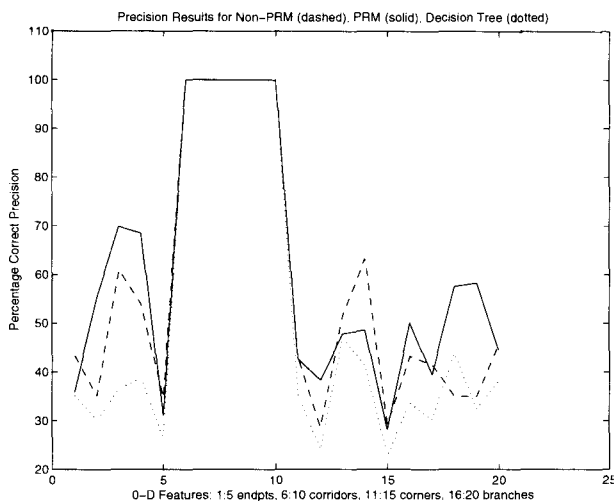


Figure 22: Percentage Correct Precision.

Image	F_endpt	F_corr	F_corn	F_bra
<i>im00</i>	35.17	100.0	35.35	33.71
<i>im17</i>	30.17	100.0	24.15	30.14
<i>im18</i>	36.42	100.0	47.14	43.79
<i>im19</i>	38.79	100.0	41.21	32.43
<i>im22</i>	26.35	100.0	22.96	38.18

Table 6. Precision Percentage Correct for Decision Tree Method.

As can be seen from the performance data in Figures 21 and 22, the static PRM feature classification method works very well with regard to recall, but requires a post-processing step to ensure good precision.

4 Conclusions and Future Work

We have demonstrated that the use of static Pseudo Range Maps yields high percentage detection of point features in engineering drawing images. We

are currently investigating:

- *Agent motion during feature recognition.* First, 0-dimensional features will be analyzed, then 1-dimensional, and finally, 2-dimensional.
- *Learning characteristics of the line drawing data.* Let the trajectory of an agent as it explores a drawing be considered an element. Then one approach to finding related parts of the drawing, or parts that are similar to a known model, is to use the affinity graph method[6] to cluster the elements into similar classes.

We are currently pursuing these two lines of development. In addition, we intend to expand the domain of application to other datasets, including: map features, handwritten and other documents.

Acknowledgments

This work was supported in part by ARO grant number DAAD19-01-1-0013.

References

- [1] F. Aurenhammer. Voronoi Diagrams – A Survey of a Fundamental Geometric Structure. *ACM Computing Surveys*, 23:345–405, 1991.
- [2] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A Solution to the Simultaneous Localisation and Map Building (SLAM) Problem. *IEEE-T Robotics and Automation*, 17(3):229–241, June 2001.
- [3] Dov Dori. Dimensioning Analysis toward automatic understanding of engineering drawings. *Communications of the ACM*, 35(10):92–103, October 1992.
- [4] D.S.Guru, B.H.Shekar and P.Nagabhushan. A simple and robust line detection algorithm based on eigenvalue analysis. *Pattern Recognition Letters*, 25(1):1–13, 2004.
- [5] Feng Su, Jiqiang Song, Chiew-Lan Tai and Shijie Cai. Dimension Recognition and Geometry Reconstruction in Vectorization of Engineering Drawings. In *Computer Vision and Pattern Recognition, 2001 (CVPR 2001)*, volume 1, pages 682–688, 2001.
- [6] D.A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, Upper Saddle River, NJ, 2003.
- [7] H. Choset and K. Lynch and S. Hutchinson and G. Kantor and W. Burgard and L. Kavraki and S. Thrun. *Principles of Robot Motion*. MIT Press, Cambridge, MA, 2005.

- [8] Thomas C. Henderson. Explicit and Persistent Knowledge in Engineering Drawing Analysis. Research Report UUCS-03-018, School of Computing, University of Utah, Salt Lake City, Utah, 2003.
- [9] Thomas C. Henderson and Lavanya Swaminathan. Form Analysis with the Nondeterministic Agent System (NDAS). In *Proceedings of 2003 Symposium on Document Image Understanding Technology*, pages 253–258, Greenbelt, VA, April 2003.
- [10] Thomas C. Henderson and Lavanya Swaminathan. NDAS: The Nondeterministic Agent System for Engineering Drawing Analysis. In *Proceedings of International Conference on Integration of Knowledge Intensive Multi-agent System*, pages 512–516, October 2003.
- [11] Thomas C. Henderson and Lavanya Swaminathan. Symbolic Pruning in a Structural Approach to Engineering Drawing Analysis. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 180–184, August 2003.
- [12] Jiqiang Song, Min Cai, Michael R. Lyu and Shijie Cai. A New Approach for Line Recognition in Large-size Images Using Hough Transform. In *International conference on pattern recognition (ICPR'02)*, volume 1, page 10033, 2002.
- [13] L.E. Kavradi, P. Svestka, J.C. Latombe, and M.H. Overmars. Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. *IEEE-T Robotics and Automation*, 12(4):566–580, June 1996.
- [14] J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.
- [15] Martin Levine. *Vision in Man and Machine*. McGraw-Hill Book Company, New York, 1985.
- [16] M.Mattavelli, V.Noel, and E.Amaldi. A New Approach for Fast Line Detection Based on Combinatorial Optimization. In *10th International Conference on Image Analysis and Processing*, volume 1, page 168, 1999.
- [17] M. Montemerlo and S. Thrun. Real Time Data Association for FastSLAM. In *Proceedings of International Conference on Robotics and Automation*, Taipei, Taiwan, 2003.
- [18] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision*. Addison-Wesley, Reading, MA, 1992.
- [19] S. H. Joseph, T. P. Pridmore, and M. E. Dunn. *Toward the automatic interpretation of mechanical engineering drawings*. Computer Vision and Image Processing(A. Bartlett, Ed.), New York: Kogan Page, 1989.
- [20] Jiqiang Song, Michael R. Lyu, and Shijie Cai. Effective Multiresolution Arc Segmentation: Algorithms and Performance Evaluation. *IEEE-T Pattern Analysis and Machine Intelligence*, 26(11):1491–1506, November 2004.
- [21] Stuart Russell and Peter Norvig. *Artificial Intelligence*. Prentice Hall, Upper Saddle River, New Jersey, 2003.
- [22] Lavanya Swaminathan. Agent-Based Engineering Drawing Analysis. Master's thesis, University of Utah, Salt Lake City, Utah, May 2003.
- [23] K. Tombre and D. Antoine. Analysis of Technical Documents using A Priori Knowledge. In *IAPR Workshop Syntactic and Structural Pattern Recognition, Pont--Mousson, France*, pages 178–189, 1988.
- [24] K. Tombre and Dov Dori. Interpretation of engineering drawings. In *Handbook of Character Recognition and Document Image Analysis*, pages 457–484, 1997.
- [25] Y.Yu, A.Samal, and S.C.Seth. A System for Recognizing a Large Class of Engineering Drawings. *IEEE Transactions on PAMI*, 19(8):868–890, 1997.

Perceptual Organization in Semantic Role Labeling

Prateek Sarkar

Eric Saund

Perceptual Document Analysis
Palo Alto Research Center
3333 Coyote Hill Road, Palo Alto, CA 94304
{psarkar,saund}@parc.com

October 24, 2005

Abstract

Documents are produced for the purpose of human interpretation. Human perceptual factors have played an important role in the design of documents — from the development of glyphs and scripts to the layout of visual components. OCR technology allows recovery of textual content from images of text but does not recover visual information encoded in layout. We explore the role of perceptual organization in the interpretation of documents and related computational challenges. A useful application area is semantic role labeling: the problem of assigning semantic roles to visual or textual structures in documents. We present a generic A^* search formulation that has been applied to reduce search times by orders of magnitude in a semantic role labeling problem.

1 What is Semantic Role Labeling

Documents, as records of human communication, are built up of visual components that play distinct roles in expressing the meaning of a document. While these roles are various, and visual artists are still discovering ways of communicating through images, major categories of semantic roles can be enumerated in the context of document processing. For example, business invoices contain visual components that communicate identity (address block, company logo), object relations (item-price relations, signator-signature pairing), special attention (highlighting, circling). Parsing a document image into its semantically significant components is the problem of Semantic Role Labeling. We develop automatic and

semi-automatic approaches to Semantic Role Labeling especially for large volume image understanding and indexing projects.

2 What is Perceptual Organization

The science of Perceptual Organization, pioneered by Wertheimer, Koffka, and Köhler, studies patterns and principles by which humans organize visual stimuli into perceptual forms in ways that are quite universal across both peoples and situations. This science advocates the Gestalt principles of perceptual grouping, and theories of figure-ground separation. The common Gestalt principles of grouping by proximity, similarity, continuity, closure, symmetry, surroundedness apply to both design and analysis of visual documents. In all scripts, proximity informs the grouping of glyphs into words. Curvilinear alignment, and feature similarity are vital in perceiving more complex textual blocks as visual objects rather than just a collection of glyphs. Enclosures and separators are frequently used to guide perceptual and semantic grouping.

3 Perceptual Organization in document image interpretation

The interpretation of complex document layout structure has remained a research subject. Unlike the parsing of one-dimensional text, image parsing even in very structured domains is algorithmically difficult, because of the lack of a natural ordering in

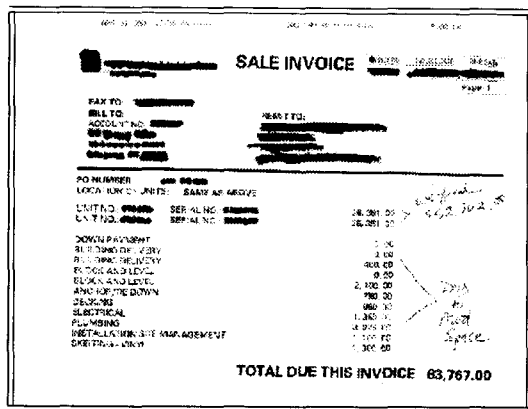


Figure 1: Perceptual queues in parsing a business invoice. Semantically significant information such as the logo and address areas, header alignment, tabular structure, presence of hand-written annotations, graphic separators are all visually salient features in the design of the invoice, and/or important cues in the analysis of such images.

two dimensions. Although very constrained grammar models (such as X-Y trees [NKS93]) can keep parsing problem tractable, parsing of images in general presents an exponential search space. This exponential complexity derives from the exponential number of ways in which image primitives can group to form larger objects. Bottom-up grouping to date has been engineered for proximity, and special cases of curvilinear continuity (e.g., horizontal sequences of words form text lines). Non-overlapping boxes and convex hulls have also been used as criteria for pruning groupings in images [MV98].

If alternative groupings are to be pruned to keep the search tractable, we believe that such pruning should be heavily informed by perceptual organization principles, especially because document designs and conventions have evolved to suit human perception. Of course, perceptual grouping cues can also directly aid to resolve ambiguity in grouping. In other words they can form a vital component of the objective function rather than just pruning heuristics. Figures 2 show examples of images where layout of image primitives rely on perceptual grouping for their interpretation. Tree grammars are in general not sufficient for capturing the compositional structure of such images.

4 Research challenges

Hierarchical decomposition of document images (top-down or bottom-up) according to preset rules can help in analysis and semantic role labeling of a limited set of documents. Documents containing mainly textual material arranged in rectangular layouts have been the main targets of successful methods. In more complex documents (tables, maps, handwritten memos, drawings, photographic images) both segmentation, and the parsing of segments into meaningful structure is, in general, exponential in the number of alternatives. Interpretation such documents will require segmentation of images, and their interpretations to guide and inform each other. The principal challenges are:

- *A model of document structure:* The logical and visual structures of documents – constituents, correspondences and relationships – will have to be expressed in a language that allows for information from one domain to be extracted into the other.
- *Machine learnable, adaptive models for visual structure:* The challenge is to build models for the Gestalt grouping principles that can be repurposed for a variety of document image analysis applications, and use such models in the parsing of document images.
- *Smart search:* The interpretation of a given image is not the output of a number of pre-defined processes, but rather a search through the space of all possible segmentations, groupings and interpretations. This search process should be adaptive to the image at hand, generate and keep alive multiple plausible interpretations, incorporate a principled way of comparing alternative hypotheses, and be prudent regarding hypotheses that are necessary to evaluate (to beat the curse of exponential explosion).

5 Conquering the search problem

We shall provide a glimpse of the formulation of a specific semantic role labeling problem, and an A^* search algorithm that we have developed for its efficient solution. Consider an example where an image has been decomposed into its compositional atoms, *i.e.*, chunks of image that can take on a semantic label

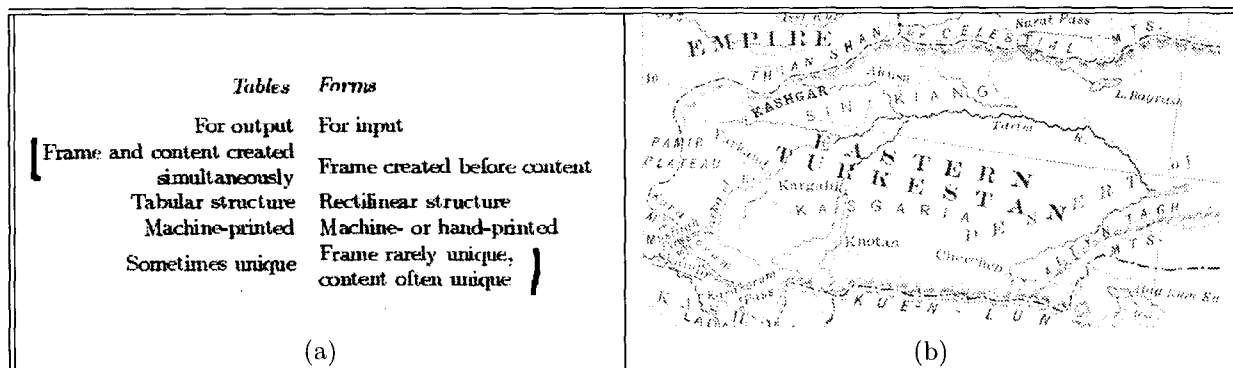


Figure 2: Examples of perceptual grouping in images. (a) The vertical alignment of text line groups is vital evidence that the marked text lines should be grouped into a single table cell. (b) The vertical alignment of text line groups is vital evidence that the marked text lines should be grouped into a single table cell. Parsing under such grouping cues is of exponential complexity.

without further subdivision. The problem is to assign labels (y_1, y_2, \dots, y_n) to a collection of observation atoms (x_1, x_2, \dots, x_n) . In a typical pattern recognition formulation, the assignment is chosen by maximizing some objective function $f(x_1 \dots x_n, y_1 \dots y_n)$ over all possible labelings. If each atom can take on one of C label-values, the label assignment problem is a search through the space of C^n label assignments. Restricting the nature of the objective function avoids the exponential explosion of the search space. For example, if the objective function is factorizable as the product of functions of the form $f_i(x_i, y_i)$, our problem reduces to labeling n observation independently, and the search complexity is nC . But often factorization of the objective function, if present, is not as simple. In addition to intrinsic features of x_i that influence the labeling y_i , we may have:

- *Factors coupling two or more labels* of the form $h(y_i, y_j, \dots)$. Linguistic constraints in OCR are in this category. A Gestalt preference for repeated structure can be modeled with such such factors. In parsing a book page, at most one of all the text-word atoms may take on the label “page-number”.
- *Factors coupling two or more observations* of the form $g(x_i, x_j, \dots)$. These arise when the atomic observations have constraints on relative size, orientation, or color change (lighting variations). Style consistency [SN05, VN05] models require such factors to model shared fonts, color, or other characteristics of text.

- *More complex factors* such as direct dependence between labels and features of different atoms. For example, to label a dot in an image as a “bullet” induces preference for indentation on adjoining atoms, and alignment with other adjoining “bullet” atoms. Both indentation and alignment are features of groups of atoms.

Most perceptual organization principles induce such complex dependencies among atomic constituents of images. Various techniques have been developed in the graphical models literature for solving the joint labeling problem (e.g., loopy belief propagation, sampling, variational methods, and hybrid techniques.) However these techniques do not offer guarantees of joint optimality. We have developed an A^* search framework, where we find easily factorizable upperbounds for complex objective functions. In particular, when the upper-bound is expressed as a product of n functions $f(x_i, y_i)$ the bound can be maximized in linear time. It is then used to guide the search in a best first way, thereby guaranteeing that the optimal solution will always be found. While the worst case complexity of this method is still exponential, the low average complexity can enable the solution of problems that are otherwise intractable.

6 Application: Indexing engineering drawings

A typical repository-indexing or metadata-extraction application requires scanned images of drawings to be indexed by a number of fields such as drawing number, drawing title, date of creation, author, company

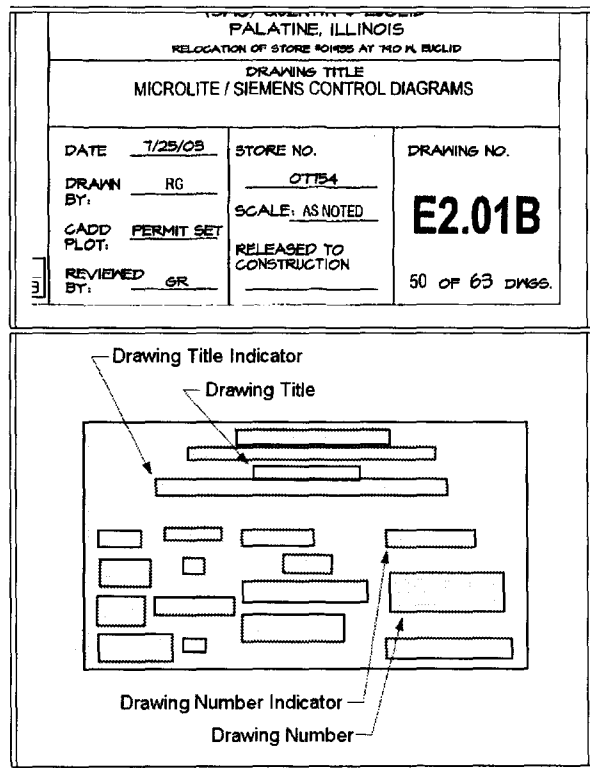


Figure 3: An example of a title-block image fragment showing the four relevant fields to be extracted, and the competing irrelevant text groups in the neighborhood.

logo, site name, and scale. In our pilot technology development project we focused on extracting drawing numbers and drawing titles. Figure 3 shows an example of an image fragment that contains the drawing number and drawing title. While most drawings include drawing numbers and titles, they are printed at various locations, and in various formats and styles (Figure 4.)

Our objective is to locate these fields in the images without restricting input to a small number of layouts and styles. To this end, an input image is pre-processed to generate a number of candidates for the drawing number and title fields. The first step is perceptual grouping of foreground pixels into potential text fields. Connected components are first grouped to obtain word-like elements, which are further grouped into text-lines and multi-line text objects. Proximity, size similarity, alignment, and surroundedness (a text group does not straddle line enclosures) are used in the grouping process. In the typical engineering drawings in our dataset, this results in 30-200 text groups (see Figure 3.) Each resulting

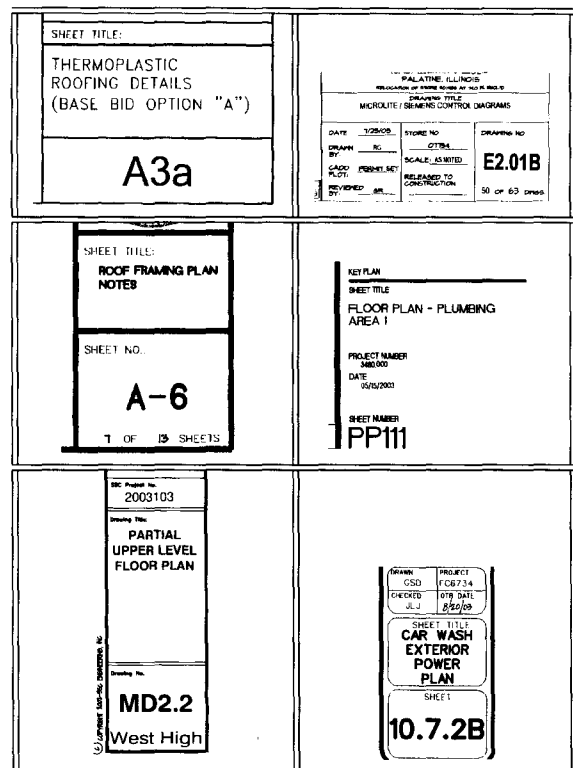


Figure 4: Examples of index fields in snippets of engineering drawings: drawing number and drawing title.

group in the image is sent to an external OCR engine, and any deciphered text is recorded.

We assume that each relevant field is grouped into a single unit, and not merged with other fields. In other words, each group found by our grouping algorithm, along with its OCR text, is treated as an atom for labeling. Our problem then is to label at most one of these groups as *drawing title* (*DT*), and at most one as *drawing number* (*DN*). To aid in the process, we also attempt to find a *drawing title indicator* (*DTI*) (e.g., "Sheet Title",) and a *drawing number indicator* (*DNI*) (e.g., "Sheet Number"). All remaining fields are to be labeled *irrelevant* (*IRR*). We thus have $C = 5$ classes, $n \approx 50$ (on average) objects to label, and a search space of $\approx 50^5$ competing label-hypotheses.

A label-hypothesis is represented as $(l_{dn}, l_{dni}, l_{dt}, l_{dti})$ meaning that the group l_{dn} is labeled *DN*, group l_{dni} is labeled *DNI*, and so on. $l_{(,)}$ can take on values $1 \dots n$ or 0, the latter indicating that the field is missing. Our problem is to find a label hypothesis that maximizes some score. The score (likelihood) of a label-hypothesis is assumed to

be of the following form:

$$L(l_{dn}, l_{dni}, l_{dt}, l_{dti}) \propto f_{dn}(l_{dn}) \cdot f_{dni}(l_{dni}) \cdot f_{dt}(l_{dt}) \cdot f_{dti}(l_{dti}) \cdot u(l_{dn}, l_{dni}) \cdot v(l_{dt}, l_{dti}) \cdot w(l_{dn}, l_{dt}) \quad (1)$$

The formula contains two kinds of functions:

- *Unary functions* f_{dn} , f_{dni} , f_{dt} , f_{dti} , which compute a goodness score (likelihood) that a group is a *DN*, *DNI*, *DT*, *DTI* respectively based on measurements that are properties of the group alone. Examples of such properties are *size*, *aspect ratio*, *position on page*, *regularity of size of group components*, *average stroke width*, *closeness of OCR text to expected textual content*, etc. In our pilot experiment we use only the last mentioned feature.
- *Pairwise functions* u , v , w , which compute scores on pairs of label candidates. These functions may thus depend on *relative location* and *relative alignment* of two groups, whether the two groups are *neighbors in a Delaunay triangulation*, whether they *share an enclosure* demarkated by line or white-space separators, etc. Pairwise functions can also be used to model mutual exclusivity constraints such as ($l_{dn} \neq l_{dt}$), *i.e.*, the same group cannot be both a drawing number and a drawing title. In our pilot experiments only the relative location of two labeled groups and mutual exclusivity were used in the pairwise functions.

In general, functions involving three or more groups can also be included. The overall score is proportional to the product of all such functions, and the winning label-hypothesis is the one that maximizes this product. If the objective function includes only unary functions, each component unary function can be maximized independently to arrive at the result. The pairwise (and higher order) terms, introduce complexity because such independent term-by-term maximization is no longer useful. Nevertheless, the pairwise functions are important for distinguishing, for example, the drawing number from the many other alpha numeric fields that appear on a page. So we follow the A^* solution suggested in the previous section, by replacing the pairwise terms by fixed upper bound to obtain an overall upperbound score of the form:

$$\hat{L}(l_{dn}, l_{dni}, l_{dt}, l_{dti}) \propto f_{dn}(l_{dn}) \cdot f_{dni}(l_{dni}) \cdot f_{dt}(l_{dt}) \cdot f_{dti}(l_{dti}) \cdot \hat{u}\hat{v}\hat{w} \quad (2)$$

This upper bound is now factorizable, and can be used in our A^* search. We can sort all n candidate text groups in an image in decreasing order of $f_{dn}()$, $f_{dni}()$, $f_{dt}()$, $f_{dti}()$, respectively. The resulting four lists contain candidates groups, ordered from best to worst according to our upper bound, for the *DN*, *DNI*, *DT*, *DTI* labels respectively. The top candidates from each list are combined to form our most-promising label-hypothesis.

For the search algorithm we construct a priority queue, designed such that the head of the queue is always the most-promising label-hypothesis yet to be evaluated. A label-hypothesis is evaluated by computing its true score according to formula 1, comparing it to the running best hypothesis (according to true score). The search is over when the true score of the running best hypothesis exceeds the upper bound score of the next most promising candidate waiting at the head of the queue.

Of course, the queue does not have to be populated with all n^4 label candidates at start. It can be grown dynamically as the search proceeds. Whenever we pop the head of the queue, we also insert four successors of the popped label-hypothesis. A successor of a label-hypothesis can be obtained by picking any element of the hypothesis, say the index of the *DN* label - l_{dn} , and replacing it with the next most promising *DN* index, easily obtained by moving down the $f_{dn}()$ sorted list by a notch.

In practice, the queue will seldom process all of n^4 possible label hypothesis, because with a good upper bound, the top label-hypothesis will be found among a few most promising candidates. In our experiments we noted that only of the order of a hundred label-hypothesis candidates were being explored among a potential n^4 candidates, where n was typically between 50 and 200. If the queue for an image grows much larger, it indicates extremely weak discriminating evidence from the unary functions. In such cases our algorithm has a high probability of being in error, even if we let it run till the end, and it may be useful to simply *reject* this input as undecipherable.

This is work in progress. In initial experiments on approximately 1000 test images, the drawing number field was correctly identified about 80% of the time. Observed errors derive from the following major causes: failures of the grouping algorithm to generate correct atomic units of text for labeling; OCR errors; errors in the unary functions to identify the textual properties of target labels.

7 Concluding remarks

Metadata tagging of document content is complementary to brute-force search over unstructured text. With the rise of XML standards for representing metadata in electronic databases, this function is gaining increased societal as well as commercial interest. Discovery of atomic content items, and the tagging of their semantic roles, is a key aspect of this problem.

We have outlined a formal approach to Semantic Role Labeling which decomposes the problem into construction of atomic units using perceptual organization techniques, and subsequent labeling of these units using unary and multiway evidence. We have demonstrated an A* approach to managing inherently exponential search, in cases where the atomic units are fixed. A greater research challenge arises when the atomic units cannot be computed with confidence, but instead multiple candidate parsings must be entertained.

This Semantic Role Labeling approach to document content tagging is amenable to other document recognition data sets and we are interested in expanding the set of domains to which it may be applied.

References

- [MV98] E. Miller and P. Viola. Ambiguity and constraint in mathematical expression recognition. In *Proceedings of AAAI-98*, pages 784–791, 1998.
- [NKS93] G. Nagy, M. Krishnamoorthy, S. Seth, and M. Viswanathan. Syntactic segmentation and labeling of digitized pages from technical journals. *IEEE Trans. PAMI*, 15(7):737–747, 1993.
- [Sar02] P. Sarkar. An iterative algorithm for optimal style conscious field classification. In *Proceedings of the 16th ICPR*, 2002.
- [SM04] E. Saund and J. Mahoney. Perceptual support of diagram creation and editing. In *International Conference on the Theory and Applications of Diagrams*, 2004.
- [SN05] P. Sarkar and G. Nagy. Style consistent classification of isogenous patterns. *IEEE Trans. PAMI*, 27(1):88–98, January 2005.
- [VN05] S. Veeramachaneni and G. Nagy. Style context with second order statistics. *IEEE Trans. PAMI*, 27(1):14–22, January 2005.
- [Wer38] Max Wertheimer. *Laws of Organization in Perceptual Forms (Translation from 1923 German article by W. Ellis)*, pages 71–88. Routledge & Kegan Paul, London, 1938.

Pictographic Recognition Technology Applied to Distinctive Characteristics of Handwritten Arabic Text

Mark A Walch

The Gannon Technologies Group
1000 North Payne Street
Alexandria, Virginia 22314
(703) 373-1962
mwalch@gannontech.com

Donald T Gantz, PhD

George Mason University
4400 University Drive
Fairfax Virginia 22030
(703) 993-1511
dgantz@gmu.edu

Abstract

Pictographic Recognition technology is a Graph-Theory-based method for locating specific words or groups of words within handwritten and machine printed document collections. This technique converts written and printed forms into mathematical graphs and draws upon key properties of graphs such as topology and geometric features to locate graphs of interest based upon specified search terms. In its initial implementation, Pictographic Recognition technology has functioned as a search tool by identifying individual characters in strings of handwritten script—both English and Arabic. Arabic, however, through its internal segmentation resulting from intra-word gaps, presents the opportunity to treat groups of characters as distinctive objects that can be recognized directly without the need to isolate individual characters. This paper focuses on current research toward using Pictographic Recognition techniques to identify Arabic word segments.

1 Introduction

Handwritten Arabic poses unique challenges for recognition technologies. First, Arabic is written in a cursive form so there is no clear separation between characters within words. Second, Arabic characters change their form depending on their word position: initial, middle, final or standalone. Third, Arabic words incorporate external characteristics such as diacritical markings. Fourth, Arabic writers take broad “latitude” with handwritten forms by stacking characters as well as omitting certain characters. This latter characteristic creates considerable dissimilarities between handwritten and machine printed forms of Arabic.

Of the items cited above, it is Arabic’s unconstrained cursive form that arguably creates the most difficult problem for recognition technologies. In conventional recognition systems such as those used for hand or machine printed Latin characters, it is possible to

“segment” words into individual characters before actual recognition begins. While it is a major impediment to machine recognition, segmentation comes quite naturally to human readers because humans appear to perform segmentation in conjunction with recognition rather than as separate sequential tasks. That is, given that a human reader has a prior knowledge of accepted character forms coupled with an understanding of context, readers identify these forms directly within a cursive word, rather than isolating them first and recognizing them later.

Aside from Arabic’s cursive form, the stacking and omission of characters pose the next greatest challenge to recognition technologies. Stacked characters work contrary to the expectation of sequential horizontal placement of letters and omitted letters negate the certainty that characters will always be found in expected locations. What human readers can readily overcome constitute significant obstacles to algorithmic solutions. These obstacles are particularly effective in barring technologies developed to recognize machine print from making a smooth transition to handwriting—in Arabic as well as other languages.

Pictographic Recognition technology offers a viable approach toward recognizing cursive forms of Arabic and English as well as “pictoform” languages such as Chinese. Pictographic Recognition shares common roots with Optical Character Recognition, but it is a fundamentally different approach. Pictographic Recognition looks for the “Pictographic Signature” of words in their native form. In this context, Pictographic Recognition is somewhat similar to Optical Word Recognition (“OWR”) but differs significantly from OWR in its ability to evaluate the actual characters and character groupings that comprise the unknown words.

To date, Pictographic Recognition has been implemented as a Prototype application that performs word searches based on pictographic signatures. These

signatures consist of graph-based topologies embedded in written or printed words. These topologies usually represent characters, but they may also reflect small character groups or parts of characters such as loops, cusps and line crossings. Current languages where Pictographic Recognition has been implemented include machine printed, hand printed and cursive English as well as machine printed and handwritten Arabic. A more detailed discussion of the basic concepts can be found in the white paper entitled: "Pictographic Matching: A Graph-based Approach Towards a Language Independent Document Exploitation Platform" which will be made available by the Authors upon request.

Aside from the focus on graphs as its foundation, another aspect of Pictographic Recognition that is distinctive from OCR is the format of output data. OCR typically converts images of documents into text. Pictographic Recognition converts images into a specially structured format that retains much of the information extracted from the original image including, candidate character possibilities as well as data related to figure topology and geometry. These data are retained so they can be applied during the actual searching process, so every item processed is evaluated directly against a search term of interest.

The interim product from Pictographic Recognition takes the form of a "Results Matrix" rather than the text results characteristic of OCR. The Results Matrix maintains considerable data about each word and character in a document collection including word and character alternatives, attendant confidence scores and related geometric information. The manner in which this information is stored can be defined as *Planned Indeterminacy*. That is, unlike OCR which "forces" its results into text output, Pictographic Recognition, refrains from word determination until an actual search is performed. At the time of the search, the full body of information stored in the Results Matrix is brought to bear to match the search term with words in the document collection being searched. In this way, the search process is "empowered" by using all the relevant data available at exactly the time it is needed. The value of these data is realized by a special search engine that incorporates Dynamic Programming techniques to find the best fit between a search term of interest and detailed word data rendered from document collections

Pictographic Recognition starts with images of documents. The images are parsed into individual objects representing words and characters within these words. Each of the parsed pieces is converted into a graph containing a wealth of physical measurements and mathematical descriptors. These measurements become the basis for a classification scheme that decodes the graph into its actual identity as a single or a

group of letters. Because of the abundance of measures available, classification can be sharpened by focusing on the specific set of measures that separates each particular letter's graph from others. As items are classified, they are loaded into a Results Matrix that retains information concerning alternative identities as well as geometric information. As searching is performed, the search term is compared against the contents of the matrix to find the best mapping of the search terms of interest with the matrix elements built from document collections.

The previous paragraphs have presented a "primer" on Pictographic Recognition. The remainder of this paper discusses the Authors' most recent research efforts related to this topic. These efforts focus on a specific aspect of Pictographic Recognition that "bundles" both segmentation and recognition activities into a single action that can be very efficient and effective in recognizing Arabic script. In particular, the discussion that follows will focus on the ability of Pictographic Recognition to treat connected multiple character strings within Arabic words as objects that can be recognized individually. At the heart of this recognition process is the ability to encode both the features and topology of a character into a form that can be used for rapid identification. The technique herein described is strongly rooted in an algorithmic approach, based on Graph Theory, which treats handwriting and printed text as mathematical graphs. The principles that have been successfully applied to individual characters will now be applied to connected character strings.

2 Conceptual Framework

Graph Theory is a branch of Mathematics that focuses on representing relationships as line diagrams containing nodal points and the linkages among these points. In graph terminology, the nodes are referenced as "vertices" and the links as "edges". Graphs are a handy and effective way to represent written language since graphs can be created directly from the pen strokes used to compose letters and words. Words, characters and numerals take their form as graphs written on paper assuming distinctive shapes representing the letters of the alphabet as well as numerals and punctuation. The edges and vertices of these written graphs connect and cross and are straight or curved. Graphs accurately capture the essence of written language since they contain both the topological structure and geometric information sufficient to replicate complete written forms. Graphs are the foundation of written language and a search methodology that focuses on the graphs offers the potential to permit searching that encompasses several languages.

Graphs contain all the information extracted from writing condensed into a concise mathematical format

that is highly computable. Within graphs, this information takes two forms. The first form is the graph's *topology* that can be seen in the connectivity among the major graph components. The way pen strokes are crossed and connected is the framework for graph topology. Topology is the structure of the graph. The second form of information contained in graphs is the *geometry* of the graph. Geometry is expressed in terms of distances, angles and characteristics of graph components. The depth or shallowness of a curve, the distance between line crossings or the sharpness of angles are all examples of graph geometry. Geometry characterizes the *shape* of the graph. Collectively, topology and geometry account for the structure and shape of graphs. The topology and geometry of graphs are also quite computable. That is, they can be expressed as data to support computer-based processes that can be performed on graphs such as indexing and searching.

The foundation of Pictographic Recognition is an algorithm that describes graph topology as a numeric code. Any two graphs with the same structure will generate the same code. Any two graphs generating the same code are said to be *isomorphic*. Linked to the graph topology and its attendant numeric code is graph geometry. Graph geometry can also be expressed as a feature vector used to compare graphs. A feature vector is a multi-dimensional expression of the multitude of measurements that can be extracted from graphs. When the topology of graphs is coupled with feature vectors as a combined data structure, the computability of this data structure offers a very effective way to use graphs as indices for characters and words contained in the images of documents.

The computational engine for Pictographic Recognition is an automated method for identifying and matching isomorphic graphs and storing these graphs in a database. Graphs are isomorphic when they are structurally identical—with the same number of edges and vertices connected in the same way—although they may appear to be different. Two graphs are considered isomorphic when there exists a one-to-one correspondence between their internal structures. That is, they have the same number of edges and vertices connected to form exactly the same topology.

Figure 1 illustrates this point. Although the graphs appear to be quite different, they are structurally identical: isomorphic. They appear different because their geometry is different. The topology is the same, but the angles and distances are different.

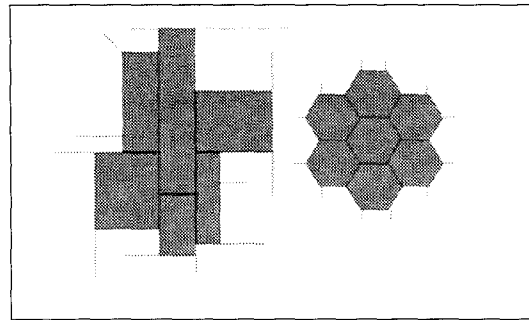


Figure 1: Illustration of two Isomorphic Graphs with different features.

Figure 2 illustrates the concept of Isomorphic Graphs as it applies to written characters. In this figure, each row shows three instances of the letter “a” written as the same isomorphic graph class. The numbers on the left hand side of the figure are the code names derived by the Pictographic Recognition process for each class of graph. Graph classes reflect graph topology. The class labeled “2;192” consists of graphs built from a loop and one edge. Class “4;112.0” contains graphs with 2 edges only. These typically characterize the letter “u” and the letter “a” written to be open at the top. Class “4;98.0.64” encompasses characters with a leading edge, an enclosed area—in graph terminology this is called a “face” --and a trailing edge.

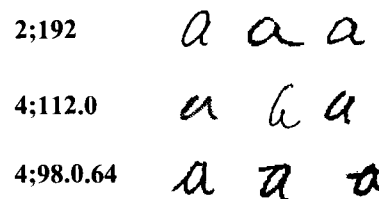


Figure 2: Sample letters “a” for three different graph isomorphic classes.

Figure 3 shows how graph isomorphic classes transcend individual letters. This figure presents letter “a” and “e” for isomorphic graphs classes coded “4;112.0” and “4;98.0.64”.

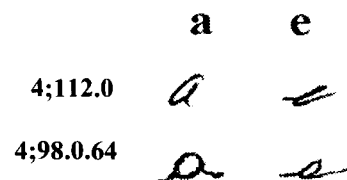


Figure 3: Example of letters “a” and “e” sharing same isomorphic graph.

Graph isomorphism is a critical concept underlying Pictographic Recognition. In fact, Pictographic Recognition employs a database that maps a collection of known graphs—such as the letters of the alphabet—against all possible unknown graphs such as those that could be extracted from imaged words. Building such a database is actually accomplished by generating all possible graphs up to a certain size (order), then testing each of these graphs to determine whether they contain the topologies of any known graphs. As these topologies are detected, a mapping is established to indicate the presence of a known character topology embedded in an unknown graph form. To construct such a database it is necessary to consider only planar graphs. Planar graphs are graphs that can only exist in a plane, such as lines on a page of paper.

In its current implementation, Pictographic Recognition focuses on locating and identifying individual characters within handwritten words. The present discussion looks at Pictographic Recognition applied in a different manner—as a means of identifying character groupings rather than individual characters. The nature of written Arabic offers the perfect opportunity to apply Pictographic Recognition to multi-character clusters.

Written Arabic is considered to be a “cursive” language. The structure of written Arabic is similar to cursive English with some significant exceptions. These exceptions include:

1. Certain characters, such as short vowels, are omitted from the handwritten form but included in the machine printed form.
2. Arabic characters change their form depending on the location within a word.
3. Rules govern which characters connect and which characters do not connect.
4. Arabic characters may be stacked vertically as well as placed in a horizontal sequence.

Figure 4 illustrates these properties by contrasting formal machine printed Arabic (above) with handwritten versions of the same words (below).

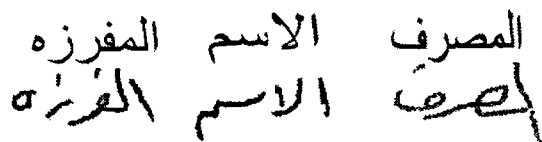


Figure 4: Comparison of machine printed and handwritten Arabic words.

Of the four distinctive Arabic properties herein enumerated, the present discussion focuses on the intrinsic segmentation of Arabic words into character

groups. These groups occur because certain Arabic characters always connect while others never connect. There is really no parallel of this concept in cursive English, except certain lead capitals that may not connect with following characters in a script word.

The “intra-word gaps” within Arabic words divide the word into smaller parts that can be treated by Pictographic Recognition in a way quite similar to the way individual characters are treated. For purposes of this paper, these segmented character strings are referenced as “word segments”.

These word segments consist of freestanding characters as well as small character groupings. The following table shows the character count distribution for word segments compiled from the Arabic Gigaword Corpus produced by the Linguistic Data Consortium. This corpus is a massive compilation of Arabic words obtained from numerous electronic media sources.

Table 1: Distribution of word segments by size from Arabic Gigaword Corpus.

Number of Characters in Segment	Number of Segments	Percent	Cumulative Percent
1	7,396,009	0.45.41	0.4541
2	4,608,726	0.28.30	0.7370
3	2,724,877	0.16.73	0.9043
4	1,077,822	0.0662	0.9705
5	334,438	0.0205	0.9910
6	104,649	0.0064	0.9975
7	17,520	0.0011	0.9985
8	23,679	0.0015	1.0000

In this table, the first column indicates the number of characters in the observed word segments. The second column shows the number of segments detected of a particular character length as shown in the first column. The third and fourth columns show each row as a percentage and as a cumulative percentage of the total. This table shows that 99 percent of Arabic word segments contain 5 characters or less. And 90 percent of the Arabic word segments contain 3 characters or less. This observation may be in part an artifact of Arabic’s “triconsonantal” roots in which vowels are inserted into consonant-based templates.

The following table provides a similar analysis from a 1,000 page handwritten Arabic document collection obtained by the Authors. Note the striking similarities between the distributions of counts of characters within character sequences between this sample and the above selection from the Arabic Gigaword Corpus.

Table 2: Distribution of word segments by size from handwritten Arabic document collection

Number of Characters in Segment	Number of Segments	Percent	Cumulative Percent
1	29,746	0.5247	0.5247
2	14,622	0.2579	0.7826
3	7,756	0.1368	0.9194
4	3,004	0.0530	0.9724
5	1,187	0.0209	0.9934
6	357	0.0063	0.9997
7	14	0.0002	0.9999
8	5	0.0001	1.0000

The fact that Arabic word segments are built from so few characters helps to constrain their complexity and makes them ideal candidates to be treated and classified by Pictographic Recognition in a manner similar to individual letters. Most Arabic words are built from 2 to 4 word segments. Focusing on these segments and treating them as individual objects to be recognized converts the problem of recognizing multiple connected characters within a Arabic word into a matter of recognizing a few individually segmented forms. However, the difficulty related to recognition based on word segments is that segments—even for the same string of characters—will not take exactly the same form and, thus, will not necessarily generate isomorphic graphs even if they represent the same string of characters.

Making use of word segments requires a method that can focus on what is common among the segments while de-emphasizing the differences. In other words, the task becomes one of finding the “essence” of word segments. Pictographic Recognition offers a means of capturing this essence. Specifically, the essence can be found in forms embedded within word segments that are significantly large to capture the principal features of any individual segment and yet simple enough to be common to several word segments representing the same set of characters. To Pictographic Recognition the essence of word segments is found in embedded isomorphic graphs.

3 Embedded isomorphic forms

Written representations of character sequences are usually quite similar graphically and distinguished only by a few differences such as extra or omitted strokes. Because of these differences, the graphs these characters produce will be different in terms of the strict definition of graph isomorphism. That is, to be isomorphic, graphs must be identical. However, there will often exist an embedded graph that transcends multiple writing samples and is isomorphic in each

sample. That is, this graph becomes a common embedded form. Figure 5 shows two versions of the English word “is”. In their native form, these words have several differences principally related to the additional strokes along the bottom of the right hand sample and the extra lines in the “dot” above “s” on the left side. However, they do have a significant common embedding that is shown in the lower portion of the figure.

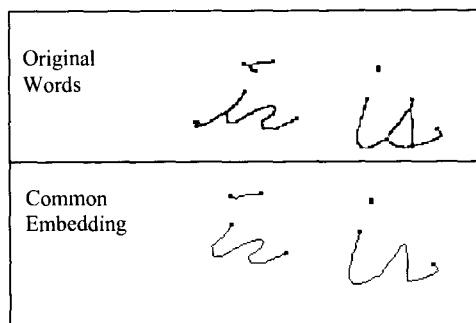


Figure 5: Comparison of original form and common embedded forms for English word: “is”.

Embedded forms offer an effective tool for capturing the essence of written character strings. Because of the great fluidity and variability of handwriting, the likelihood of two handwritten strings or words matching is small. However, these different strings could well have much in common in terms of embedded forms. Arabic word segments behave in a manner similar to cursive English word strings. As such, Arabic word segments contain embedded graphs that are common among the segments. Figures 6 and 7 provide some examples to show these embeddings.

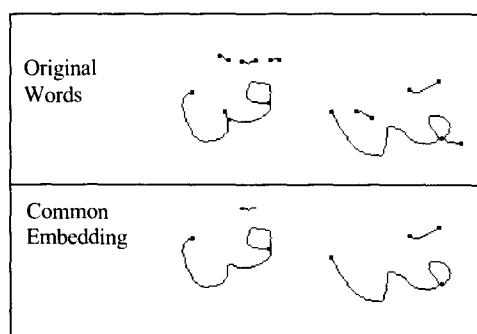


Figure 6: Comparison of original form and common embedded forms for Arabic word قن

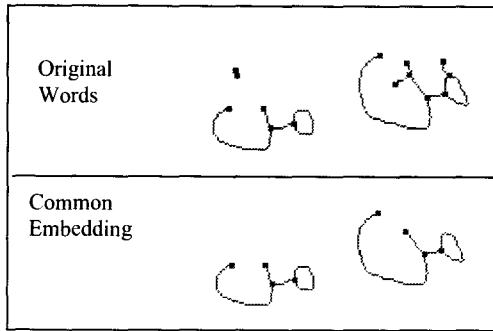


Figure 7: Comparison of original form and common embedded forms for Arabic word من

4 Conceptual Foundation for detecting Embedded Isomorphic Graphs

Treating written objects as graphs presents computational opportunities to detect common embedded forms. Graphs existing in one plane, such as graphs produced from written forms on paper are considered to be “planar graphs”. As such, planar graphs are a subset of all possible graphs. Furthermore, within the collection of planar graphs, a database can be built considering all the non-isomorphic forms of such graphs. The order of a graph indicates the number of vertices (or nodes) in the graph. The following table shows the number of non-isomorphic planar graphs up to order eleven.

Table 3: Distribution of isomorphic graphs up to order 11.

2	1
3	2
4	6
5	20
6	99
7	646
8	5,974
9	71,885
10	1,052,805
11	17,449,299

This table indicates there are 17,449,299 possible unique graphs up to and including graphs of Order 11. All graphs of order lower than 11 are also included in this amount. Thus, any possible graph up to this size can be pre-calculated and all instances of known character graphs can be mapped to these pre-calculated models. While 17 million is a large number, it is a very manageable and very computable number by present computing and storage standards. Review of graphs existing in language indicates Order 11 is a reasonable model both for English and Arabic.

An evaluation of character graphs contained in the reference data provided by the National Institute of Science and Technology shows that for written English

over 99.5 percent of characters as actually written create graphs of Order 11 or less. Similarly, analysis of a database of cursive English shows over 99.5 percent of all written characters produce graphs of less than or equal to Order 11. Experience indicates a higher percentage for Arabic because of innately simpler character forms. The fact that there are 17,449,299 theoretically possible graph forms that will contain virtually one hundred percent of all known graph forms provides the practical foundation for building the mapping database.

Creating this database presents another challenge, however. And creation is possible through an indexing technique that assigns a unique code to each unique graph isomorphism. That is, each graph with edges and vertices connected in a unique way can be assigned a unique numeric code to express its singularity of structure. Detailed discussion of how this code is derived is beyond the scope of this paper, but it involves arranging a graph’s adjacency matrix into a particular state. Adjacency matrices are tabular representations of the connections of vertices by edges within a graph. If graphs are isomorphic, it is possible to arrange their adjacency matrices to match exactly. So, all the adjacency matrices of all isomorphic graphs can be arranged into exactly the same order and this order can be “hashed” into a numeric value. Thus, a key can be built that will reflect each unique graph topology and this key can be used as access into a database of all possible graphs up to a certain size. The calculations to build this database are considerable, but they need only be performed once and stored as data. The best discussion of building isomorphic graph keys can be found in the body of the U.S. Patent Application entitled: “Systems and Methods for Source Language Pattern Matching” (Walch).

The concept of the isomorphic graph database was invented to store information regarding individual characters. However, the concept readily extends to character groups such as n-grams and word segments. In the case of Arabic word segments, each segment is treated as a graph and stored in the database. Traditional segmentation in the sense of horizontal “cutting” of the word segment is not necessary. Rather, the task becomes one of finding the “essence” of the word segment—the embedded form that characterizes an individual word segment and many other word segments resulting from the same character combinations.

Although graph isomorphism is a very important ingredient for capturing embedded graph forms, it must be stressed that isomorphism alone is insufficient for recognizing these written forms since similar topologies may result from completely different characters. The graph’s feature vector, as measured by angles, distances

and other descriptors, must also be considered in concert with the graph's topology. Within a set of graphs of the same topology (isomorphic graphs), these graphs can be classified based on their physical characteristics. In the case where these graphs represent the letters of the alphabet, classification separates the letters "a" from "e" even though the underlying graphs are isomorphic. Again, refer to Figure 3 which shows different letters constructed from isomorphic graphs.

Features to support graph classification cannot be mathematically divined. Rather, features for classification must be obtained by modeling from exemplars obtained from actual writing specimens. Thus, classifying Arabic word segments is a two-step process. The first step involves a modeling stage where a classification algorithm is trained to distinguish isomorphic graphs that represent different character combinations. The second step is to apply this classifier to unknown word segments to be recognized.

5 Modeling Arabic Word Segments

Building a model to classify Arabic word segments requires that a suitable training set of graphs be established. This set of graphs should contain the various graph forms generated for different character combinations as well as common embedded forms. The modeling set can be accomplished through the following steps.

1. Obtaining a set of Arabic writing samples
2. Isolating and labeling the individual word segments within these writing samples.
3. Converting these word segments into graphs.
4. Extracting the common embedded isomorphic graphs from the word segment graphs for each character combination.
5. Training the classifier using the word segment graphs and principal embedded graphs.

Steps 1 and 2 are pretty much straightforward. They involve collecting writing specimens and labeling the character sequences within these specimens. The most important aspect of these two steps is to obtain a collection of writings sufficiently broad to encompass character sequences likely to be of interest in future searching.

Two strategies will achieve this objective. The first strategy is to train on a listing of likely key words from an authority list. That is, given *a priori* knowledge of likely terms of interest, collect handwriting specimens containing these words. The second strategy is more complex. This strategy involves building a language model containing words that provide the broadest coverage of the language. One way the magnitude of this task can be approximated is to select a sampling of words from a large collection of words in common

usage, such as the Arabic Gigaword Corpus, and extract and compare word segments with a specific document collection.

A sampling of words with a frequency greater than 500 drawn from the Gigaword Corpus will yield a list of 3,172 words. These words will produce 1,777 unique word segments. Compared against a collection of handwritten Arabic language documents containing 32,507 words and 56,961 word segments, the segments extracted from the Gigaword Corpus match 87 percent of the words from the document collection. This suggests that simply building a training set from an attainable collection of commonly used words offers a viable approach for collecting a comprehensive set of Arabic word segment exemplars to support word-segment-based recognition.

Once the samples have been collected, the next step entails converting the word images into graphs. This step is followed by another step that involves extracting common embedded sub-graphs that consistently characterize the word segments. These embedded isomorphic graphs can be extracted directly using the isomorphic database concept. As a word part is encountered, it is converted into a graph and its isomorphic key is generated. This key serves as the basis for looking up the graph in the isomorphic database. Once the key lookup is complete, the isomorphic database returns a list of all embedded sub-graphs.

The isomorphic database will identify all embedded sub-graphs based solely on topology. As such, these graphs may not actually represent the corresponding components of word segments.

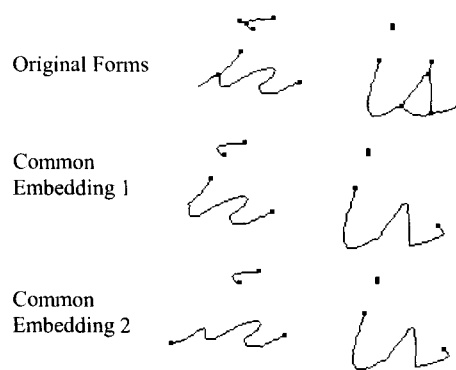


Figure 8: Comparison of Original and two embedded forms for handwritten word "is".

Figure 8 shows two possible combinations of embedded sub-graphs for handwritten samples of the English word "is". Whereas the original graphs are not isomorphic, the isomorphic database extraction method found two

pairs of common embedded graphs that were isomorphic to each other. However, the first set of embedded graphs provides a better match to the actual corresponding parts of the original words.

Finding the embedded isomorphic graphs that best match the original words can be accomplished by filtering techniques rooted in geometric features. That is, as embedded graphs are extracted, their geometric features can be compared to find those offering the closest match. These geometric features include lengths of graph segments, directions among graph components, location of “corners” and other shape-related features and the like. The result of this process is to establish “clusters” of similar embedded graphs for Arabic word segments. It should be noted that a segment may produce several different clusters based on the variations of writing used to generate the word segments.

6 Classifying Graphs

A graph feature vector consists of the wealth of measurements that can be obtained from graphs. These measurements include directions, distances and various descriptors. Directions are quantified as angles between graph components such as the angle from one vertex to another or the angle from the centroid of an edge to a vertex. Similarly, distances quantify the amount of space between graph components such as the number of pixels between two vertices. Graph descriptors include factors that articulate the shape of graphs. Two such measures are Bezier values, which provide a means for capturing the essence of curves in as few as four numbers and Bending Energy that is an indicator of curvature. Figure 9 shows three classes of features that are part of a character graph’s feature vector.

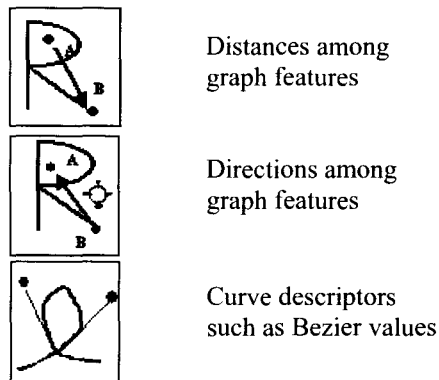


Figure 9: Three principal classes of features used for Pictographic classifying of graph forms.

When isomorphic graphs are aligned, a direct one-to-one correspondence between each geometric measurement comprising the feature vector can be

established. That is, when two graphs are identified through the isomorphic graph matching process, their feature vectors are also aligned and corresponding features can be compared in detail.

Figure 10 shows graph alignment between two separate embedded graphs from Arabic word segments for the Arabic word من . These segments were extracted from different words from different writers.

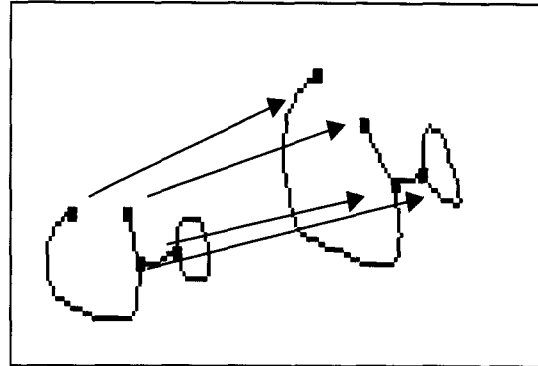


Figure 10: Alignment for two embedded forms for Arabic word من taken from different writing samples.

The isomorphic graph matching process ensures that the two versions of the Arabic word “من” are matched and aligned. Alignment means that all vertices are matched in corresponding pairs as indicated by the arrows in the figure. Given the point-to-point alignment achieved by isomorphic graph matching, graph feature vectors can be aligned to reflect the corresponding relationships between two graphs. The process of matching graphs and aligning the geometric features in the feature vector is a very robust technique for recognizing varying forms of characters and symbols when other techniques—principally derived for recognition of printed fonts—are much less forgiving of these variations in form. Once topologies are matched and feature vectors aligned, the graphs are ready to be modeled as individual objects that represent both single and multiple character strings.

Considerable data are available from graph-based feature vectors. Even a relatively simple graph can generate a vector with over a hundred measurements. This wealth of data is well suited for a modeling method utilizing Stepwise Discriminant Analysis. Discriminant Analysis is a powerful multivariate statistical technique that can be used to identify the subset of variables that best distinguishes one graph from another. Using Stepwise Discriminant Analysis, head-to-head comparisons can be made between aligned feature vectors to identify the specific set of measures that distinguishes one object from the other.

Modeling for ultimate classification through Discriminant Analysis involves considering each set of isomorphic graphs separately. Each set of graphs may represent different character strings but have the same underlying topology. During this modeling process, a specific set of variables is isolated for each form of each character set within the isomorphism. This feature set constitutes the "Alphabetic Kernel" for the word segment. The Alphabetic Kernel is defined as the specific set of measures that will reliably distinguish that a particular word segment from all other segments represented by the same isomorphic graph.

Similarly, a Regression Tree Classifier can be used to isolate those features that distinguish one character string representation from another. The Classifier distills the abundance of measurements to a succinct set of critical measurements that distinguish one form from another. In either case, the result is a set of powerful variables labeled by the Authors as the Alphabetic Kernel.

Through the modeling process, Alphabetic Kernels are generated for all forms of all word segments extracted from the training data. These kernels taken in concert with the specific isomorphism provide the basic data to decode unknown word segments into their appropriate text character strings.

7 Recognition of Arabic Word Segments

Recognition of Arabic word segments can be accomplished through a process that parallels the method used for modeling. This process is characterized by the following steps.

1. Extract objects likely to be word segments from handwritten documents.
2. Convert these segments into graphs.
3. Generate an isomorphic database key from the word segment graphs.
4. Use the key to find the graph in the isomorphic database.
5. Extract the significant embedded graphs.
6. Classify these graphs based on the training data.

The application of Pictographic Recognition to word segments differs slightly from its application to individual characters. The principal difference is that individual character identification requires a preliminary form of segmentation that breaks the word into small individual parts that can be assembled in different ways to capture the actual characters. In the case of the word segments, advantage is taken of the intra word gaps as the means of segmentation. In both cases, the output can be handled the same way.

When applied at the individual character level,

Pictographic Recognition distinguishes itself from OCR by retaining its results in a matrix that retains alternative candidate results. Since word segments will resolve to character sequences, results derived from Arabic word segments can be handled and stored in the same way.

8 Preliminary Recognition Results

A preliminary test was structured to measure the effectiveness of Pictographic Recognition in classifying individual segments within Arabic words. This test involved a training set consisting of 120 Arabic single paragraph writing samples each containing 109 words. Collectively, these words contained 247 word segments. Another set of 120 handwritten Arabic documents containing the same paragraph but from different authors was used as the testing set. In both cases, the documents were manually labeled at the word and character level to provide ground truth both for training and testing purposes.

The training documents were automatically segmented into words and word segments using the existing functionality of the current Pictographic Recognition software platform. The word segments were further processed automatically to identify common embedded isomorphic graphs. A threshold was set to extract only those embedded graphs that occupied more than 75 percent of the graph structure of the original word segment from which it was extracted. As embedded graphs were extracted, they were labeled by their appropriate isomorphic graphs. A Regression Tree Classifier (CART® "Classification and Regression Tree" by Salford Systems) was used to build classification trees for the different embedded isomorphic graph groups extracted. For each isomorphic graph group, the classification tree contained the salient variables and decision rules to distinguish the various character combinations associated with the graph. The number of unique character combinations within each isomorphic graph group ranged from as few as ten to over one hundred. This process is identical to the manner by which individual characters are also handled for classification by Pictographic recognition. Tables 5 - 7 show examples of Arabic word segment classification for three isomorphic graph groups. The graph groups shown in these tables encompass the sets shown in Table 4.

Table 4: Profile of selected graph groups.

Graph Group Code	Training /Testing Sample Size
6;96.128.16	1,930
4;64,128	3,155
6;112.0.16	2,453

Table 5: Classification results for top nine word segments within graph group 6;96.128.16.

True Class	Count	Percent Correct	CLASSIFIED AS									
			None	تما	ثر	سر	سما	كل	لمز	لو	مع	يد
None	1510	67.61	1021	37	35	18	10	45	89	33	63	58
تما	29	89.66	2	26	0	0	0	0	0	0	1	0
ثر	48	97.92	1	0	47	0	0	0	0	0	0	0
سر	34	82.35	4	0	0	28	0	0	0	0	0	0
سما	45	77.78	7	0	0	0	35	1	2	0	0	0
كل	54	92.59	4	0	0	0	0	50	0	0	0	0
لمز	60	93.33	3	0	0	0	0	0	56	0	1	0
لو	39	79.49	3	0	0	0	0	0	0	31	0	0
مع	63	90.48	2	0	0	0	0	0	4	0	57	0
يد	48	91.67	2	0	0	0	0	0	0	1	1	44

Table 6: Classification results for top nine word segments for graph group 4;64,128.

True Class	Count	Percent Correct	CLASSIFIED AS									
			None	أ	س	سا	شا	لسا في	ها	خو	يد	
None	1958	67.67	1325	12	70	38	118	110	17	48	127	93
أ	264	96.59	4	255	0	2	0	0	0	1	1	1
س	115	93.04	4	0	107	1	0	1	0	0	2	0
سا	112	89.29	1	0	6	100	0	0	0	5	0	0
شا	87	85.06	6	1	1	1	74	2	0	1	1	0
لسا في	79	82.28	9	0	2	0	2	65	0	0	1	0
ها	123	99.19	1	0	0	0	0	0	122	0	0	0
خو	83	86.75	4	2	1	0	2	1	0	72	1	0
يد	232	89.22	15	0	0	0	5	4	0	0	207	1
None	102	84.31	10	0	0	0	1	0	0	0	5	86

Table 7: Classification results for top nine word segments for graph group 6;112.0.16.

True Class	Count	Percent Correct	CLASSIFIED AS									
			None	أ	شا	لسا	نب	هر	خو	يد	ير	ين
None	1728	73.73	1274	56	74	32	46	78	21	17	27	103
أ	90	72.22	20	65	1	0	0	0	1	3	0	0
شا	73	83.56	9	0	61	0	1	0	0	0	0	2
لسا	81	96.30	1	0	2	78	0	0	0	0	0	0
نب	52	78.85	5	0	1	0	41	0	0	0	0	5
هر	64	84.38	8	0	0	0	0	54	1	0	1	0
خو	63	92.06	3	0	0	0	0	1	58	0	0	1
يد	145	84.83	5	3	0	0	0	1	0	123	6	7
ير	53	84.91	4	0	0	0	0	1	0	1	45	2
ين	84	72.62	15	0	2	0	3	0	0	2	1	61

The size of the testing sample was intentionally set to the same quantity as the training sample. These three isomorphic graph groups were selected from the hundreds of possibilities generated because they are among the most frequently occurring graphs for Arabic word segments.

Figure 11 shows sample Arabic characters that produce the three graphs used in the analysis. It should be noted that individual isomorphic graph groups transcend both individual characters as well as word segments.







Graph Class	Example	Example 2
6;96.128.16		
4;64.128		
6;112.0.16		

Figure 11: Examples of graphs applied to Arabic characters.

For each of the three sets of graphs, training was performed using only the top nine character strings based on frequency. This limit was imposed to test classification of items not within the training set. Testing was performed with many more character strings to determine how items not in the training set would be handled during classification. The total number of unique word strings in each isomorphic group is shown in the following table.

Table 8: Number of items processed and accuracy for three selected graph group.

Group	Number of Word Segments	Number Classified
6;96.128.16	122	9
4;64.128	111	9
6;112.0.16	119	9

In Tables 5-7, the leftmost column in each shows the top nine character strings for each isomorphic graph group. The tenth category, labeled "None" is used for those cases where a test item did not match any of the nine selected character strings. The "Count" column shows the number of items for each word segment. The "Percent Correct" column shows the percentage of test items that correctly classified. The remaining

columns show the distribution of classification results. The total numbers of items processed and percentage of correct answers are shown in Table 9.

Table 9: Number of items processed and accuracy for three selected graph groups.

Group	Count	Percent Correct
6;96.128.16	1,930	89.04
4;64.128	3,155	90.89
6;112.0.16	2,433	83.12

In addition, the groups scored 67.67 (Group 6;96.128.16) , 67.61 (Group 4;64.128) and 73.73 (Group 6;112.0.16) in terms of correctly identifying items that were not among the nine possible character sets that could be classified. These results show considerable promise for applying Pictographic Recognition to Arabic word segments.

In practice, Pictographic Recognition employs a scoring methodology that creates a composite score from the individual classification results of all embedded sub-graphs extracted from an Arabic word segment. The classification results shown in Tables 5 - 7 provide a view of how three of these embedded graphs would perform.

Figures 12 - 14 show some sample Arabic word components and the composite scores they produce. The scores are tabulated as the sum of the classification scores of individual embedded graphs. Below each image, the top 5 scores are shown with an asterisk ("*") placed next to the score of the correct answer.



Word Segment	Score
1 ط	788*
2 ل	391
3 ح	388
4 لها	226
5 ها	181

Figure 12: Sample character graph form and top five classification scores for Arabic word segment ط . This graph had ten embeddings.



	Word Segment	Score
1	نـب	1086*
2	فـي	333
3	قـن	314
4	فـي	291
5	بـ	285

Figure 13: Sample character graph form and top five classification scores for Arabic word segment نـب. This graph had 20 embeddings.



	Word Segment	Score
1	بـا	277
2	بـا	252*
3	حـا	159
4	هـا	137
5	بـا	136

Figure 14: Sample character graph form and top five classification scores for Arabic word segment بـا. This graph had 2 embeddings.



	Word Segment	Score
1	نـو	890*
2	بـط	573
3	بـه	239
4	بـ	230
5	نـو	202

Figure 15: Sample character graph form and top five classification scores for Arabic word segment نـو. This graph had 12 embeddings.

In these examples, three of the figures had the correct answer in the first position with a broad spread between first and second position. In one case, (Figure 13) the correct answer came in second position closely following the first position score.

9 Conclusions

Word segments intrinsically formed in Arabic words offer an extraordinary opportunity toward both word recognition and word detection. Pictographic Recognition provides a tool capable of recognizing either the actual word segment or common embedded graphs within the word segments. The common embedded graphs are a critical ingredient toward successful word segment recognition because the high degree of variability in handwriting reduces the likelihood that written character strings will regularly appear exactly alike. However, the embedded graphs offer a means of capturing the “essence” of these character strings in a physical and mathematical form that can reliably be captured and classified. Effectively, the embedded graphs offer a means of standardizing strings of characters into reliable consistent forms.

Initial findings indicate Pictographic Recognition shows promise for recognizing word segments both in terms of accuracy of recognition and reduction of false positives. The embedded graphs appear to capture the essence of the written character strings from which they are extracted. However, implementing a general-purpose search and recognition system requires compilation of a database with sufficient exemplars to cover as many word segments as possible. Complete coverage is not necessary, because Pictographic Recognition can always rely on its character-based processes to analyze Arabic word segments never encountered previously.

Immediate next steps include greatly expanding the collection of word segment exemplars used for training to achieve a broader overall Arabic language coverage. In conjunction with the expanded training sets, the Authors will continue to refine the techniques herein discussed toward the goal of building an accurate and reliable Arabic word search and recognition tool based on Pictographic Recognition.

Document Analysis Applications

A Document Triage System for Army Application

Francis Fisher, Kelvin Marcus, Richard Chang, Joi Turner

U. S. Army Research Laboratory
2800 Powder Mill Road, Adelphi, Maryland

Abstract

Military campaigns routinely put Soldiers in contact with foreign documents that they are generally unable to read. The PDA-based Basic Language Translation System (PDA BLTS), an ARL designed handheld document triage system, permits the Soldier to triage, or sort by importance, foreign language documents. The system assists the Soldier in the collection and triage of captured documents by identifying those documents relevant to the Soldier's mission or the overall military campaign so that the Soldier can expedite the processing of those documents. Through document triage the military linguist or translator is provided those documents that may be most relevant to military operations. BLTS was designed in response to the need for expedient document triage and delivery through a portable system to be used by Soldiers. This paper will describe the engineering methodology used to develop this tool.

1 Introduction

PDA BLTS is a handheld document triage system consisting of a laptop, five PDAs with PDA-mounted cameras, and two scanners, as shown in Figure 1. The BLTS PDA gives the Soldier a handheld device to capture and translate document images in order to determine the importance of captured foreign language documents. The translated document and keyword search results are returned and displayed on the BLTS PDA. The BLTS Server receives document images from multiple BLTS PDAs, processes those document images, and returns the triage results to the appropriate BLTS PDA. The BLTS Scanners (flat-bed and sheet-fed scanners) attach directly to the BLTS Server and provide an additional method of document input for translation and review. Documents processed by BLTS can be reviewed on the BLTS Server using the BLTSDocumentReview software.

The design goal for PDA BLTS was to develop a portable, handheld device to be used by Soldiers for document translation and triage. Before beginning the design process several problems were outlined and discussed in order to determine practical yet optimal solutions for hardware and software. Those issues included system architecture, PDA selection, scanner selection, camera selection, and software design.

This paper will describe the technical work performed, engineering methodology used, and the choices made in developing the BLTS system. It will cover what we did to test, evaluate, and solve problems, based on what problems were already known; what new problems appeared and how they were handled; and how the system was integrated.

2 System Engineering

The first question to answer was how to meet the user's need for a small hand-held document triage device while working within the limitations of available technology. While there are some small X-86 compatible tablet computers available now, we were unable to find suitable products for the BLTS application when this effort was started. Even the current X-86 based tablets are much larger than typical PDAs. If we limit the Soldier-carried device to a PDA then we seriously limit the processing

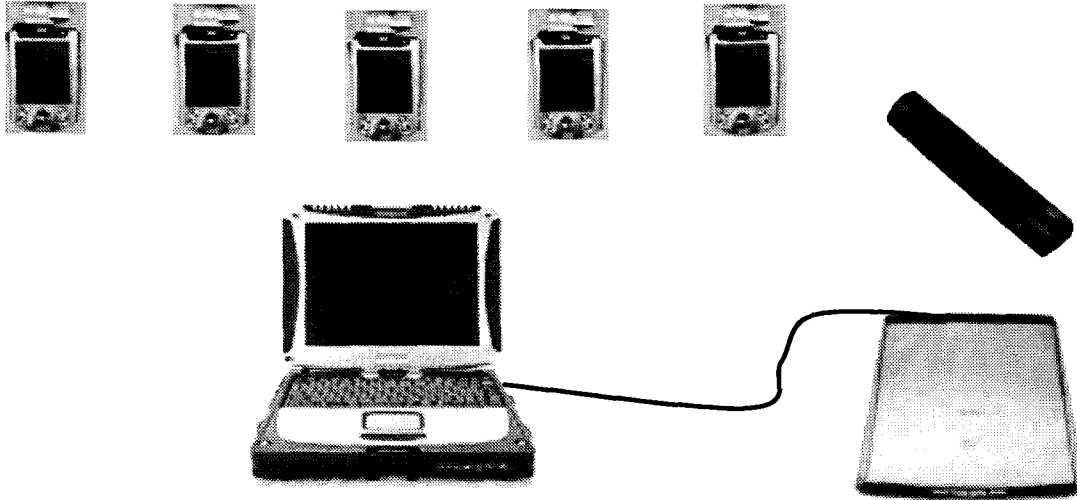


Figure 1: PDA BLTS Components

BLTS Process Flow

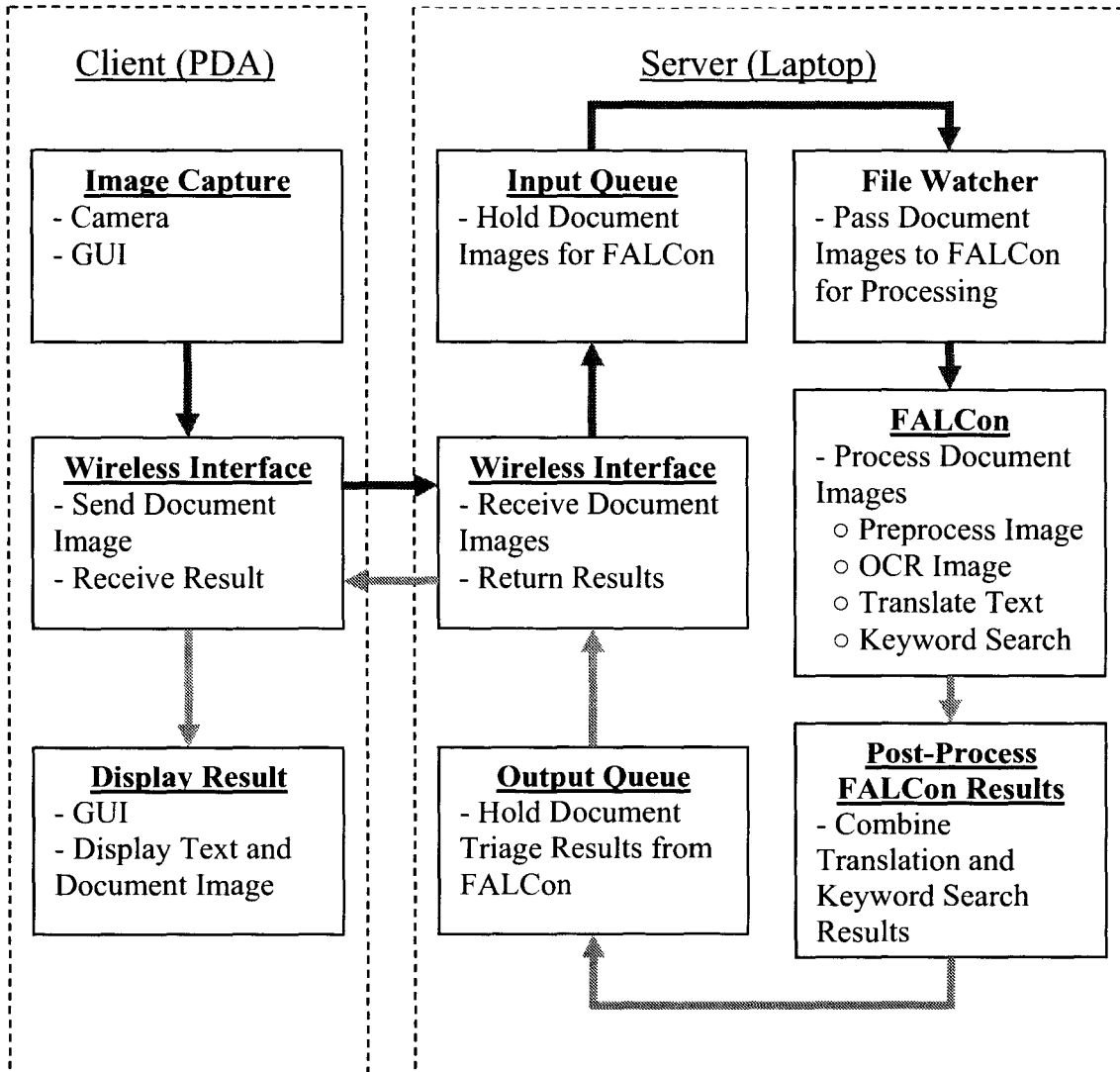


Figure 2: Process flow for PDA BLTS

capability that is available. We decided to use a client/server architecture with a laptop for the server and PDAs as the clients. This permitted us to use the existing FALCon document triage software to provide the core functionality. The PDA is then used as a document capture and display device. A wireless network is used to transfer document images from the PDA to the laptop, and to transfer the translation result from the laptop back to the PDA for review. The process flow for PDA BLTS is shown in Figure 2.

The next issue then was to select an appropriate method for document input for the PDA. Using any type of scanner would dramatically increase the size of the handheld system and would not be feasible. We decided instead to use a digital camera module that would plug into the PDA client device. Based on initial work using digital cameras to input document images for translation [1] and on test data collected in the lab, we knew that VGA resolution cameras were simply not sufficient for document image processing unless the area to be translated was severely limited. The release of 1.3 mega-pixel cameras was the final system component that ensured the success of the PDA BLTS system design.

Another issue in the design of PDA BLTS was the limitation of the wireless data link. If the Soldier operates the BLTS PDA beyond the range of the data link then document images cannot be sent to the BLTS Server for processing. Due to the processing limitations of the PDA, we found it difficult to implement an automated queuing system for delivery of document images to the server for processing. This could also have confused the user if document images sent for processing were not returned in a reasonable time because they were being held on a queue. Instead, we put the burden of operation on the user. We provide a message to the user when the wireless link is not available and they are given the option of saving the image for retransmission at a later time.

With the system engineering solutions in hand, we then turned to selecting hardware components for PDA BLTS as described in the following sections.

2.1 Hardware Selection

In the design of PDA BLTS we were required to select suitable hardware components for all of the BLTS functions. All hardware selections were performed in conjunction with Engineering Systems Solutions of Frederick, Maryland, the integration contractor selected for this project. All of the PDA BLTS components needed to be operational in extreme field environments. Since this is a prototype for pilot field-testing, there was not a requirement that the components be MIL qualified.

In designing BLTS, we have worked toward a solution that gives the Soldier a handheld device for document triage. Due to the PDA's limitations of low processing speed, small memory capacity, and the incompatibility with COTS software developed for the Microsoft Windows XP/2000 operating systems, we were forced to utilize a client server architecture for PDA BLTS for the current system designs. Given the time constraint, there was no way to port the COTS (MT, OCR, etc.) that form FALCon to the PDA, financially or technologically. FALCon was an in-house product that could easily be integrated into the current system design, and would be completely accessible by PDAs through a client/server architecture.

2.1.1 PDA Selection

As part of system design we were required to decide on commercial versus rugged hardware platforms. There are many rugged PDA platforms available on the market today. The problem is, we will not know what will be available tomorrow. The problem is even worse in the commercial PDA market where product lifetime is typically less than two years. An additional selection criterion was the cost of the equipment. Rugged PDAs are typically 3 to 5 times more expensive than the commercial equivalents. Users were questioned on their preference for quantity versus ruggedized and their preference was for a larger quantity of commercial systems for initial pilot testing. The PDA selection criteria were: COTS product; fastest processor available; SDIO memory slot for camera; operate in extreme environmental conditions.

After purchasing and testing sample PDAs, we selected the HP h5555 for version 1 of PDA BLTS. When we started production of version 2 we found that the h5555 had been discontinued. We then re-evaluated available products and selected the hx2750 PDA by HP. The hx2750 provided a 330% increase in battery life and doubled the range of operation for the wireless network.

2.1.2 Laptop Selection

The requirements in the selection of the laptop for the BLTS server were to run the Forward Area Language Converter (FALCon) software (the core translation software for BLTS), to provide wireless network connectivity to the BLTS PDAs, and support operation at temperatures from 30° to 130° Fahrenheit. The laptop selected is the Panasonic Toughbook CF-18. Built with Centrino technology, the CF-18 runs software for the BLTS Server, scanner, and FALCon and supports wireless network connectivity for five PDA's. The server queues documents from the PDA and scanner, passes the documents to FALCon for translation and keyword search, and returns the translated results to the appropriate PDA as shown in Figure 2. If the document processed is from the scanner, then the FALCon results are stored on the server for later review.

2.1.3 Scanner Selection

The selection criteria for the BLTS scanner included small size, extended temperature operation, and simplicity of operation. From previous prototype fielding of FALCon, we found that users had difficulty using sheet fed scanners to process bound or stapled documents in the field. Based on those issues we started the initial research on the application of digital cameras for document input as reported in [1]. Flat bed scanners have been greatly reduced in size in the 7+ years since our prototype fielding of FALCon. For BLTS, we selected the smallest COTS flat bed scanner that we could find. We then tested three samples of that scanner at 130 degrees F during the day and 30 degrees F at night to simulate harsh environments encountered in some military operations. Although the temperature range tested far exceeded the manufacturers specifications, the scanners tested performed well. Based on this testing we selected the Canon LIDE80 flat bed scanner for the BLTS application.

During user training for PDA BLTS, users requested sheet-fed scanners to replace the flat bed scanners. The sheet-fed scanners are much smaller than the flat bed scanners making them more portable. The trade off is that the sheet fed scanners are not able to scan bound and oversized documents. Version 1 of PDA BLTS was delivered with flat bed and sheet fed scanners, version 2 of PDA BLTS was delivered with sheet fed scanners only.

2.1.4 Digital Camera Selection

The BLTS PDA Camera selection criteria was, for the most part, geared toward finding a camera with the highest possible resolution that would provide high quality images at close range with minimal optical distortion. The market for high-resolution compact flash or Secure Digital (SD) PDA cameras is very limited. Of the small number of devices available, we reviewed data sheets and selected four different models for evaluation. One of the cameras selected for evaluation included a photo flash capability, which we thought might be helpful. Unfortunately, the addition of the flash unit to the camera resulted in a package that was so large and so heavy that it was not at all suitable for this application. The poor quality of the optics further reduced the usability of this camera. The Veo Phototraveler and the HP Photosmart turned out to be the same cameras with a minor difference in the layout of the camera body. Both of these cameras provided good imaging capability at close range with minimal optical distortion. We selected the HP Photosmart mobile camera, a 1.3M pixel SD camera, because it provided good quality document images and was compatible with the h5555 PDA. We used the software development kit from Veo with the HP Photosmart camera to provide a full solution for the integration of the digital camera into the BLTS PDA and software.

Through previous work with 3.3M pixel digital cameras done in [1], we developed a preprocessing algorithm for document images to improve FALCon OCR accuracy. During an additional evaluation of a 5.1M pixel camera we discovered an improved method in our algorithm to increase OCR accuracy for both low and high-resolution cameras. We currently use this preprocessing algorithm on document images captured by the current 1.3M pixel SD camera before passing them to FALCon for translation.

2.1.4.1 Document Image Size Limitations

In order to evaluate the PDA camera, tests were also performed to determine the optimal range of font sizes and page sizes that would yield the highest OCR accuracy when capturing and processing document images with PDA BLTS. These tests confirmed that document images with font sizes greater than 10pt and 250 DPI yield the optimal OCR results, with accuracies from 92% - 95%. OCR accuracy fell quickly when document images with font sizes less than 10pt were processed as shown in figure 3. The same was true for document images with 320 DPI as shown in figure 4, but the drop in overall OCR accuracy was much smaller for font sizes of 10pt and less as would be anticipated given the increased number of pixels on each character.

Based on these results, the camera selected for PDA BLTS should be sufficient for capturing and processing half-page document images and should produce optimal OCR results for documents with font sizes of 10pt or larger.

3 System Software Components

The following sections describe the software components of PDA BLTS.

BLTS Server

The BLTS Server software acts as a link between the BLTS PDA and the Forward Area Language Converter (FALCon), the back-end software that produces all machine translations and keyword results. The BLTS Server keeps track of all connected BLTS PDA's. It is also responsible for receiving all

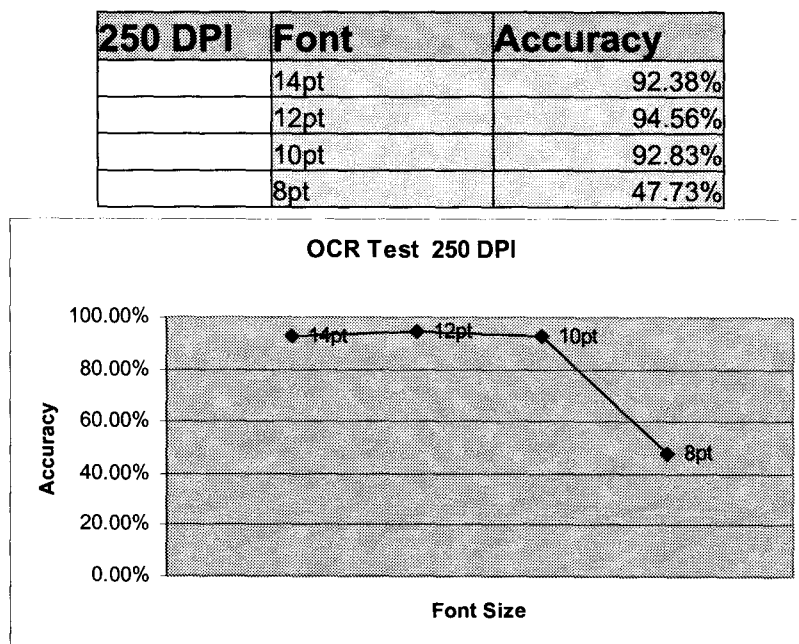


Figure 3: OCR Font Test result for 250 DPI documents

320 DPI	Font	Accuracy
	14pt	95.59%
	12pt	93.41%
	10pt	94.37%
	8pt	85.93%

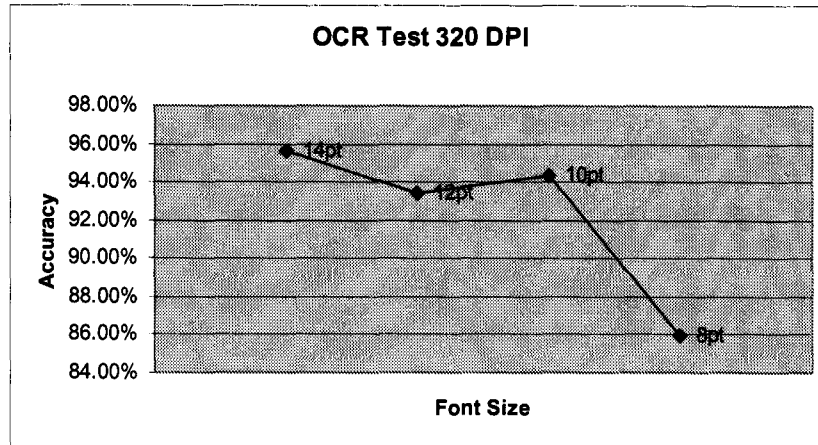


Figure 4: OCR Font Test result for 320 DPI documents

document pictures and sending all translation results to and from the PDA's. If the Soldier is near the BLTS Server with documents in hand, he or she can scan the documents, which are then automatically routed to the BLTS Server through the BLTS Scanner software. The images are then stored and translated for review.

3.1 BLTS PDA

The BLTS PDA software uses wireless communication to exchange data with the BLTS Server. The software provides a user interface to capture document images using the camera and displays the translation results for review.

3.2 FALCon

FALCon, the Forward Area Language Converter, is a tool that utilizes optical character recognition (OCR) and machine translation (MT) to help non-linguists identify the importance of foreign language documents captured in the field. Scanned document images or digital camera images can be loaded into the FALCon software which then converts the images into text characters and translates the text into English. Keyword lists can be created and specified by the user, which would be used to search the resulting English text, or if the appropriate foreign language lists exist, search the foreign language text.

During the development of BLTS, the original build of the FALCon software had undergone several key changes. The Machine Translation software, Transphere's Apptek, used by FALCon for Arabic text translation was updated from version 2.6.17 to 2.7.0.35 to address key issues where during transliteration some English word letter combinations would consistently err, word definition selection would translate to the least probable selection, and proper names would not completely translate.

3.3 BLTS Document Review- Server

All images and resulting documents can be reviewed using the BLTSDocumentReview on the laptop. This software provides a display that conveniently groups the document pictures with their respective Optical Character Recognition (OCR), Machine Translation (MT), and Keyword search results. A scaled down version of this software is also used to review MT and Keyword search result files on the BLTS PDA.

4 Software Design

The software design issues were numerous. The first issue was to figure out how to allow the system to support multiple users, and process documents concurrently. Each BLTS Server supports up to 5 BLTS PDAs; so, 5 Soldiers can send documents for processing at the same time. Since FALCon processes one document at a time, FALCon became the bottleneck in the PDA BLTS process. The solution was to adapt a multi-threaded server architecture with a document queuing process. This allowed users of the five BLTS PDAs to send documents simultaneously and to continue capturing and sending document images for processing. Controlling document return flow was the next issue to be resolved. An output queue was implemented on the BLTS Server to hold translation results. When translation results are available for a BLTS PDA on the server, the server checks to see if the PDA is available. If the PDA is not available the results are stored in the queue. If the PDA is available, then the results are returned to the PDA for review by the Soldier.

4.1 Software Improvements

The initial design for BLTS used FALCon as the backend translation software, but the processing time for this first implementation was very slow (two and a half minutes to process a half page document). The process time includes the image transmission from the PDA to the server to the moment the user is notified on the PDA that the result is ready for review. By testing the individual software components, we discovered that FALCon's initial sequence of language library loads for machine translation was contributing heavily to the drag on process time. Each time the FALCon process was called, the MT libraries would be reloaded on the server, causing unnecessary delay in the overall performance of BLTS. However, when FALCon was run in batch mode, the initialization process was only run once, at the beginning of the batch. The first document processing time was still over two minutes, but the remaining documents were processed at a rate of less than forty-five seconds. We had the FALCon team modify FALCon so that the batch mode would remain in memory and monitor a folder for new documents to translate. As a result each half-page document takes approximately 45 seconds to process.

In response to user feedback, the Soldier is given an option on the BLTS PDA that forces the server to store all translation results for that PDA on the server. The Soldier can then process larger quantities of documents without interruption and retrieve all of the results for review at a later time. Also from user feedback, documents and results are stored in non-volatile memory on the PDA in case of battery failure or system malfunction so that results will not be lost.

The first version of PDA BLTS did not take into consideration the cumbersome task of system configuration. The user had to set the network SSID, IP address and WEP encryption for the server and all five PDAs. During the initial training of the system, we noted that this section took the longest to train the Soldiers. The requirement to configure the network and security components detracted from regular system use. Because of the frequency in the need to configure the PDA, it was necessary to automate this process. The current configuration only needs a system number, which is located on the device, as user input. The network and security settings are automatically configured and re-configured, on the PDA, if that information is lost due to battery failure or a hard reboot.

4.2 User Interface Design

Another major design area was the user interface. Through many field tests and exercises we have found Army Soldiers to be both intelligent and resourceful. If you design a good product that they can use, they will use it. In all aspects of the design of the user interface we attempted to make system operation simple and flexible within the limitations of the hardware, development environment, and operating systems used. Most of the BLTS PDA software features are controlled by menu bar buttons across the bottom of the PDA display. These buttons change as the operating mode changes. All of these "soft" buttons are linked to the corresponding hardware button located under the menu item. In this way the Soldier can operate the system without the need to locate, remove, and use the stylus. The users fingertip can generally manipulate those windows that do require touch screen use for proper operation. Along with design considerations that encompassed physical attributes of the system, we had to keep in mind that the main purpose for the BLTS PDA is data management. Document information obtained from the BLTS process had to be formatted in a display so that users could make decisions upon review.

4.2.1 BLTS Document Review

The Soldier's main objective when using PDA BLTS is to aid human translators by making key decisions to ensure critical documents are identified and passed to the appropriate authorities as quickly as possible. As such, the Soldier must understand the importance of the system's output in relation to their mission. FALCon creates four document outputs during the triage process; the document image, OCR, MT, and keyword results. Each document has its importance to the end user and PDA BLTS is designed so that access to the results is simple, convenient, and understandable.

4.2.1.1 FALCon Output Format

The document image is the original view of the document either scanned or photographed. The image by itself is a digital link to the original document. By viewing the image, the Soldier can associate it to the physical document without knowledge of the language through simple pattern matching. The OCR - Optical Character Recognition output is not as helpful to the Soldier because it is not easy to see differences between the actual document and the OCR'ed document. The OCR'ed document may be more useful to the human translator who can correct OCR errors and obtain better translations. The appearance of the MT results could be best described as "word soup", or a mixture of non-cohesive words, phrases, and foreign character strings. This document can aid in the Soldier's ability to easily identify nouns, proper nouns, and action verbs and provide additional information not included in the keyword list. The keyword result document displays the frequency of mission related keywords found in either the machine translated document or the OCR document. FALCon allows the Soldier to edit and add customized keyword lists based on the current mission requirements.

4.2.2 BLTS Document Review-Server

BLTSDocumentReview is a server side application that aids users stationed at the laptop to view all four FALCon output files, as shown in figure 5. All documents scanned or transmitted from the PDA's are stored, organized on the server and accessible from BLTSDocumentReview. This application allows users to review data from a specific PDA source or the scanner. The user can determine the time and date that each data input file was captured and can also traverse through all of the input files in order. Most important, each document image is linked to its processed results. When all documents have been analyzed, the folder containing the mission's data is then closed and tagged with pertinent information regarding the mission. This data can be sent to higher commands for further investigation.

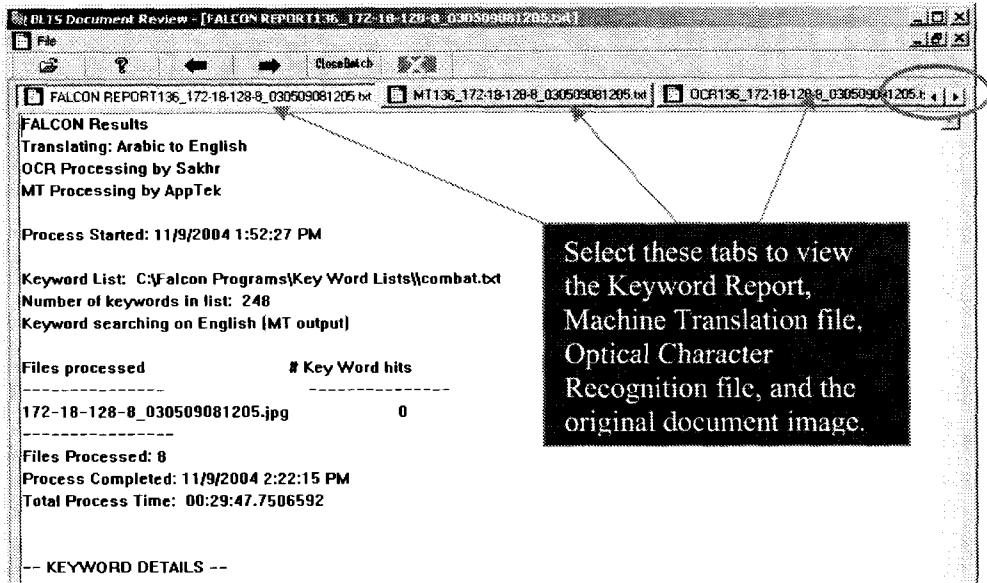


Figure 5: BLTS Server Document Review

4.2.3 BLTS Document Review-PDA

Document review on the BLTS PDA was an interesting problem. The display on the PDA is quite small and has low resolution so it is difficult to display a great deal of visual information. Since the latency between sending a document for translation and receiving the result could be more than a minute, we felt that it was very important to provide the user with the ability to see both the resulting text translation and the original document image. Our initial implementation required the user to open the text translation result from Windows File Explorer for review, then close that view and open the document image to see what the original document was. The user then had to close all views and close File Explorer to continue. This was very cumbersome. After considering different alternatives we implemented a dual view display, shown in figure 6, that shows the original document as a thumbnail view at the top of the screen with the text translation result below. If the user taps on the document image it is enlarged to fill the screen and a limited amount of text is displayed. If the user taps the document image again the display reverts the original thumbnail view with large text area. In this way the Soldier can see both the original document image and text result on one display.

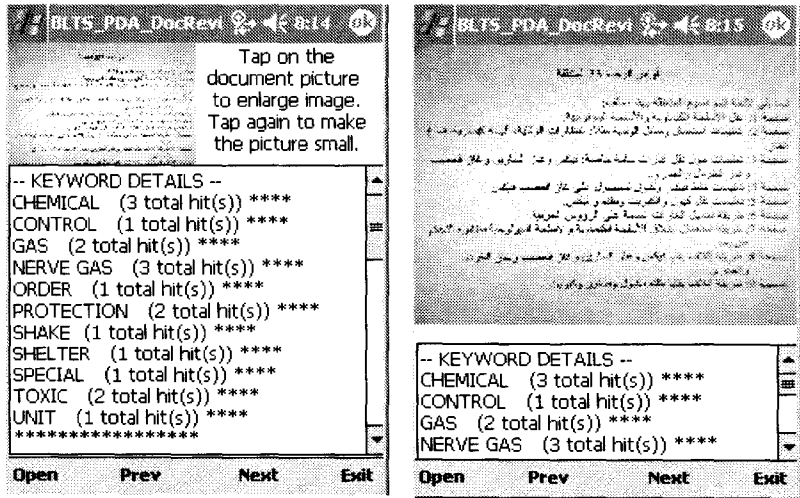


Figure 6: BLTS PDA Document Review. Machine Translation can be viewed by scrolling down in the text view.

5 Conclusion and Future Work

The various COTS software components required to implement a document triage system on a PDA were not available when this effort was started. Handheld PCs were also not available. Hence, to develop a handheld document triage system required the implementation of a client/server architecture system that utilizes a PC-based server and PDA-based handheld devices. Handheld PCs may soon be small enough to permit a handheld document triage system to be build that is based on FALCon or similar COTS based products. As an alternative to “waiting for the COTS market”, the Army is using a Small Business Innovative Research (SBIR) contract to fund the development of custom software to perform document triage on handheld PDA platforms.

Systems developed for field use must take the needs of the Soldier into account. User interfaces must be simple and easy to use, not because the Soldiers are not able to work with complex interfaces, but because the Soldiers have a larger mission to perform. We cannot anticipate that our system will receive the users full attention and we must design accordingly.

The version of PDA BLTS described in this paper will be replaced in the future by a document triage system that requires only the PDA and PDA mounted camera. This future version is already under development by contractors to the U.S. Army and some of that work will be presented at SDIUT'05.

6 Acknowledgements

We would like to acknowledge the assistance of the U.S. Army Program Manager – CHIMS for providing funding and program management support for this effort. We would like to thank ARL's Multilingual Team for providing modifications to the FALCon software as needed to support the proper operation of BLTS. We would also like to thank Dr. Michelle Vanni, Mr. Gideon Bass, and others working with Dr. Vanni, for their efforts to improve the Arabic machine translation software used in the PDA BLTS application.

7 References

[1]F. Fisher. Digital Camera for Document Acquisition. *In Proceedings for Document Image Understanding Technology*, pages 79-81, 2001

Multi-Language Handwriting Derived Biometric Identification

Donald T. Gantz, PhD John J. Miller, PhD

George Mason University
Fairfax, Virginia

Mark A. Walch

The Gannon Technologies Group
Alexandria, Virginia

Abstract

This paper provides a discussion of key technologies underlying the development and implementation of a Handwriting Derived Biometric Identification system. Three applications for author identification through handwriting are in development based on this system:

1. Individual Identification
2. Document Clustering
3. Forensic Document Examination

Individual Identification involves relating a document of unknown authorship to a data base of reference samples of handwriting from known authors.

Document Clustering encompasses grouping documents based on handwriting characteristics without knowledge of specific author identity.


Forensic Document Examination involves building a statistical foundation that will support court room testimony by expert witnesses that will withstand Daubert Challenges raised against handwriting analysis as evidence.

This paper focuses on the technical foundations of Individual Identification.

Handwriting Derived Biometric Identification exploits the rich set of measurements available through Isomorphic Graph Matching which is a technique based on Graph-Theory that is used to identify the same written forms in different writing samples. By statistically comparing measurements on similar objects across different writing we are able to identify those writing characteristics that best distinguish or characterize individual authors. An author's biometric identity is defined through the measurements that are determined to characterize that author's writing in the sense that those measurements have the power to distinguish the author's writing from that of other authors. Handwriting Derived Biometric Identification is a computationally intense process that utilizes statistical discrimination algorithms.

1 Overview

An automated process parses a handwriting sample to identify individual characters. This process uniquely associates each parsed character with a graph. For purposes of this paper, the software tool that makes this association is referenced as the "Graph Builder". Each graph is a collection of nodes connected by loops and curves. Nodes are located at the ends of curves or where curves cross. The following example demonstrates that there is a very large set of physical measurements defined for even the simplest graphs.

Character:	a	
Graph Model:	4;112	
Number of Edges:	3	
Number of Vertices:	4	

Measurements for Graph 4:112	Number of Measurements
Absolute Distance	
Vertex to Vertex	6
Vertex to Edge Centroid	12
Vertex to Edge Contours	12
Edge Centroid to Edge Centroid	3
Edge Contours to Edge Contours	3
	36
Centroid Distance	
Vertex to Graph Centroid	4
Edge Centroid to Graph Centroid	3
Edge Contours to Graph Centroid	3
	10
Graph Direction	
Vertex to Vertex	18
Vertex to Edge Centroid	24
Vertex to Edge Contours	24
Edge Centroid to Edge Centroid	6
Edge Contours to Edge Contours	6
	78
Centroid Direction	
Vertex to Graph Centroid	12
Edge Centroid to Graph Centroid	9
Edge Contours to Graph Centroid	9
	30

Table 2 presents the same style of breakdown for another author. Table 3 presents the number of times that letter/graph pairs were observed in the first author's two held out London letters. Again, frequencies in Table 3 are constrained in that frequencies are only tabulated for those letter/graph pairs observed to occur at least ten times in the author's two held out London letters. We can use the tabulated information in these style tables to investigate how strongly the patterns of letter/graph associations identify unknown authors.

Hypothetically, consider Author A's two held out London letters to be written by an Unknown Author. We can get expected frequencies if Author A were the Unknown Author by multiplying the number of occurrences of each letter of the alphabet by the decimal values represented by the percentages, as seen in Tables 1 and 2, in the row for that letter. We

can then compare the resulting expected letter/graph pair frequencies to the actual observed letter/graph pair frequencies for the Unknown Author. An obvious metric to use for measuring the differences between the expected and observed frequencies is the sum over all table entries of the squared differences between observed and expected values. We did these calculations for a set of 100 authors from our database; that is, we compared the pattern of observed letter/graph pair frequencies for Author A's two held out London letters to the modeling London letter data for each of the 100 authors. The resulting 100 sums of squared differences (after standardization) are presented in the histogram of Figure 1. In Figure 1, Author A is represented by the symbol '1' and each of the other 99 authors is represented by the symbol '0'. Author A, the true identity of the Unknown Author, is very clearly discriminated from the other 99 authors.

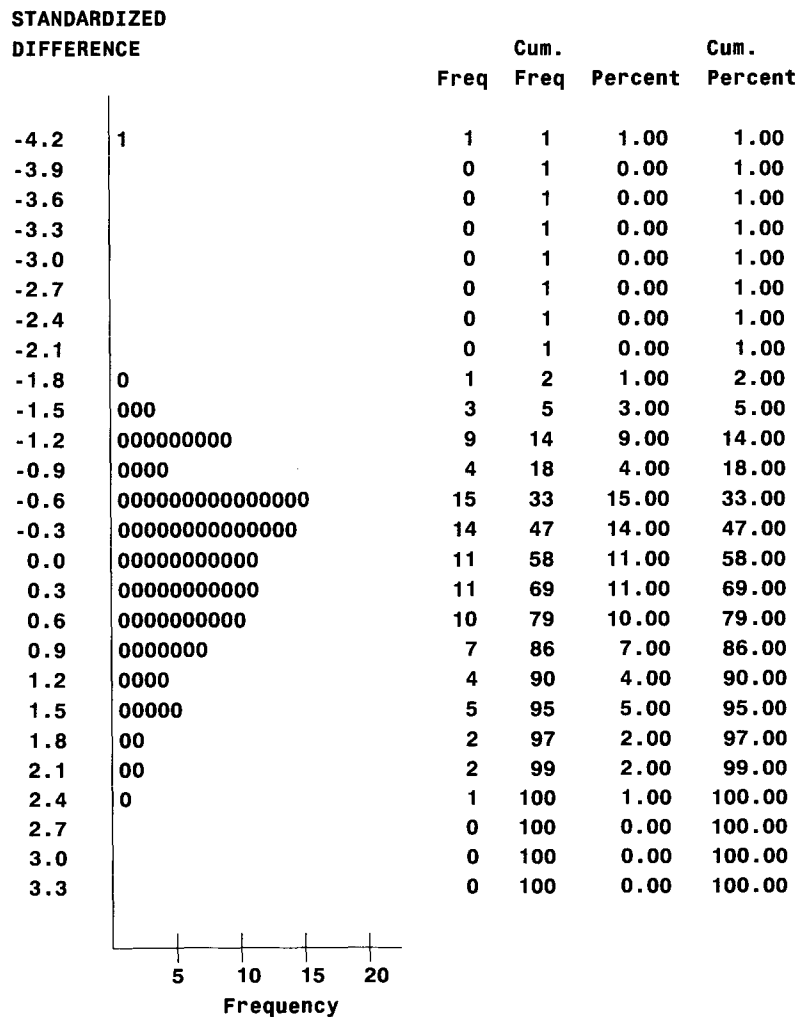


Figure 1: Standardized Sums of Squared Differences

This strong discrimination was typical regardless of the author playing the role of the Unknown Author. Each of the 100 authors had the lowest sum of squared differences among all authors when that author played the role of the Unknown Author.

The above computations show that:

1. Authors are very consistent relative to the associations between letters of the alphabet and the graphs assigned to them.
2. When the writing samples from authors in a database are as rich as they are in our London letter database, the patterns of letter/graph associations can be a powerful identifier of authorship.

3 Biometric Identification

The paper will now turn to the problem of author identification for authors who are observed to be very similar in that the Graph Builder assigns the same graph to their writings of a particular letter. The graph in question, as noted above, will have hundreds of measurements defined for it. The quantity of writings available from the author will determine the number of occurrences of the letter/graph pair. Typically, there will be many more graph measurements than there will be observed occurrences of the letter/graph pair. This makes the data analysis subject to *the curse of dimensionality*.¹

3.1 Discriminant Analysis

One of the statistical methods we use to distinguish the data for similar authors is stepwise discriminant analysis.² We apply this technique by isolating corresponding letter/graph pairs from different handwriting specimens. That is, we inspect different writing samples to locate instances of the same letter written as the same graph, assigned by the Graph Builder. In practice, this task is accomplished by automation that uses recognition to identify the character and uses the Graph Builder to assign the appropriate graph. Once the pairing is done, discriminant analysis will find a small subset of the graph measurements that do a good job of

¹ In this article, the curse of dimensionality refers to the fact that the amount of data available (in terms of occurrences of a letter/graph pair) is insufficient to support an analysis with such high dimensional data (i.e., such a high number of measurements). Some technique must be employed to reduce the measurements to a small set (the Biometric Kernel).

² SAS/STAT Users Guide, Version 9.1.

discriminating the data on this letter/graph pair for two authors. Stepwise discriminant analysis will

1. Select a small number of measurements for discriminating the two authors' data; and
2. Provide a weight (canonical coefficient) for each selected measurement.

We combine the values of the selected measurements using the weights; the resulting new measurement (canonical variable) gives the best available discrimination of the data from the two authors. Figure 2 presents a histogram for such a canonical variable and shows the resulting separation of the data from two authors.

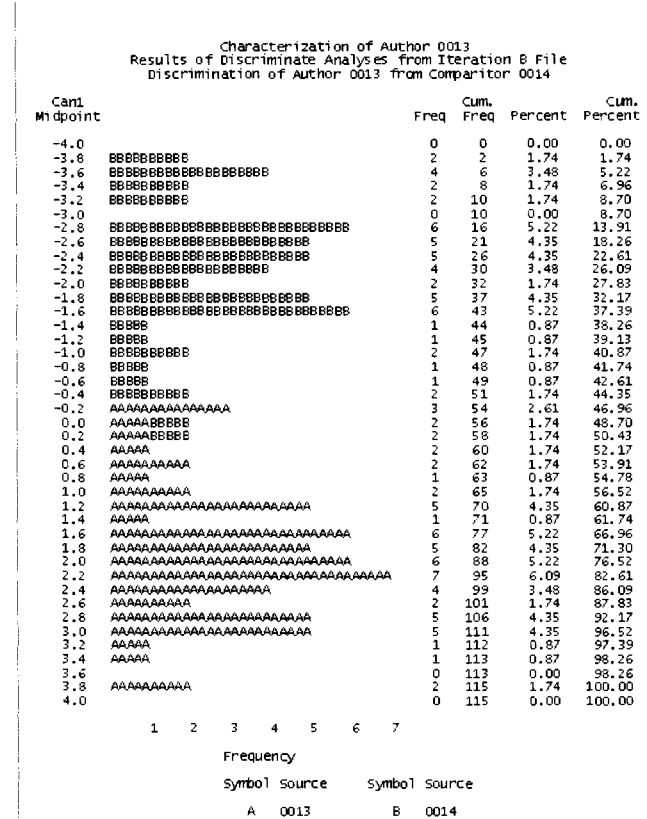


Figure 2: Discriminant Analysis for Two Authors

3.2 The Biometric Kernel

Say that we have 100 authors in our data base who have been observed to be very similar in that the Graph Builder assigns the same graph to some of their writings of a particular letter. We can use stepwise discriminant analysis to compare Author 1 one-by-one to each of the other 99 authors. We combine the results of the 99 stepwise discriminant analyses to reduce the original list of hundreds of measurements defined for the letter/graph pair to a still large but much smaller list of measurements with demonstrated power in discriminating Author 1 from

each of the other 99 authors.³ By repeating the entire process several times, we are able to isolate about a dozen of the measurements that are powerful for discriminating Author 1 from each of the other 99 authors. We refer to these final dozen (or so) measurements as Author 1's *Biometric Kernel* for the particular letter/graph pair.

Following the steps in the preceding paragraph for each author in the pool of similar⁴ authors, we define a Biometric Kernel for each author. We form a database for identification by storing the following information for each cohort of similar⁵ authors:

1. The names of the Biometric Kernel measurements for each author.
2. For each pair of authors (say, Author A and Author B), the weights (canonical coefficients), using Author A's Biometric Kernel, that form a canonical variable that discriminates Author A's data for the specified letter/graph pair from Author B's data for the same pair; and the corresponding weights associated with discrimination of Author A's data from Author B's data using Author B's Biometric Kernel.
3. The means and standard deviations of Author A's data and Author B's data for each canonical variable computed in #2.

3.3 The Competitive Matrix

We conceptualize the stored information for a cohort of similar authors as defining a *Competitive Matrix*; see Figure 3.

		Comparator Author				
		B1	B2	B3	B4	B5
Modeling Author	A1					
	A2					
	A3					
	A4					
	A5					

Figure 3: The Competitive matrix

³ The results are a subset of the graph measurements with weights; each comparison to a new author provides a new subset of measurements and associated weights.

⁴ Authors are similar in that in their three London Letters used for modeling our Graph Builder assigned the specified graph to some of their writings of the specified letter.

⁵ See Footnote 4.

The structure of a Competitive Matrix is determined by associating the matrix's rows with the author whose Biometric Kernel is the basis of the discrimination; we say that rows are determined by the *Modeling Author*. Each column is then associated with a single *Comparison Author*, that is, the author whose Biometric Kernel is not used as the basis of the discrimination. Otherwise stated, the matrix uses each author twice, once as a model author and once as a comparator author. In this way, it is possible to distil those measurements that characterize each author-to-author comparison. Since the same authors exist both as rows and columns, the diagonal axis of the matrix would contain cells where authors are compared against themselves. Since discriminating an author's own data from itself is meaningless in the current context, cells along the diagonal of the matrix are excluded from consideration.

The physical database of three modeling London letters for each author is modeled by a collection of Competitive Matrices. For the examples of this paper, a database of 100 authors is used. Each Competitive Matrix corresponds to one specific letter/graph pair. There is a Competitive matrix for every letter/graph pair observed to occur via use of character recognition and the Graph Builder application.

4 Testing the Model

We now discuss testing our Competitive Matrix modeling of author characteristics by using the Competitive Matrices to assign probable authorship to the characters in the two held out London letters for all authors in the database.

4.1 Competitive Matrix Voting

Given a character of hypothetically *unknown authorship* from the held out London letters,

1. Quantification Scheme: Associate the character with a Letter and Graph pair by character recognition and the Graph Builder.
2. Evaluate potential authorship *among similar authors in the data base*, i.e., among all authors who were observed to use the referenced Letter/Graph pair in their three London letters which were used for Biometric Kernel modeling. The evaluation is accomplished via *Competitive Matrix Voting*.

In Competitive Matrix Voting, each author plays the roles of both row (i.e., *Modeling*) author and column (i.e., *Comparator*) author; hence the matrix structure.

Each cell (intersection of a row and column) is assigned a vote, either 1 or 0 for the row author and a

vote, either 1 or 0, for the column author. The voting logic is presented in Figure 4.

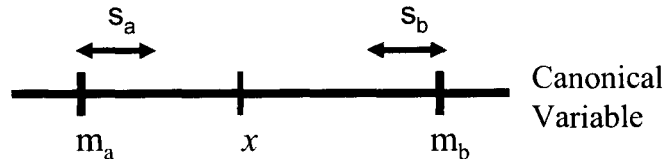
Competitive Matrix Testing a character of un

- H_0 : A is the True Author
- Model: based on assumption H_0 is true
 - So use the Biometric Kernel for Author A
- H_{alt} : B is the True Author
- Statistic is Canonical Variable for (letter/graph) pair based on the Biometric Kernel for row Author A.
- Voting Algorithm: if $|z_a| \leq |z_b|$ and $-2 < z_a < 2$ then Vote for A
if $|z_b| \leq |z_a|$ and $-2 < z_b < 2$ then Vote for B

It is possible that neither A nor B receives a vote. We call such a situation a 'no-vote' case.

$$z_a = (x - m_a) / s_a$$

$$z_b = (x - m_b) / s_b$$



m_a and m_b are the means of the data for Author A and Author B.
 s_a and s_b are the standard deviations of the data for Author A and Author B.
 x is the canonical variable (statistic) value for the character.

		parator				
		B1	B2	B3	B4	B5
Modeling Author	A1					
	A2					
	A3					
	A4					
	A5					

Figure 4: Competitive Matrix Voting

The next step summarizes each author's row votes and column votes.

Row Voting Using the Competitive Matrix with m Authors

- For Author A, Add the votes for A across all columns (all comparator authors)
- This amounts to $m-1$ tests of H_0 : *A is the true Author*
- One test for each column Comparator Author B
- Each failure to reject the H_0 with a plausible value for x is a vote for A
- An average vote close to 1 means that the unknown data is more consistent with the data used to model Author A than with the data from comparator authors.

Column Voting Using the Competitive Matrix with m Authors

- For Comparator Author B, sum the votes for B across all rows (all modeling authors)
- This amounts to 1 test of H_0 : *A is the true Author* for each modeling Author A
- Each rejection of H_0 in favor of H_{alt} : *B is the Author* with a plausible value for x is a vote for B
- An average vote close to 1 means that the unknown data is more consistent with the data for author B than with the data used to model most other Authors

As one's intuition would expect, an author's row and column votes are correlated; that is, both row and column votes are low or both are high. Typically, we have observed that the true author will have a fraction of row votes close to 1 and a fraction of column votes close to 1. However, as seen in Figure 5 some authors other than the true author might also have a fraction of row votes close to 1 and a fraction of column votes close to 1. In Figure 5, each author from the Competitive Matrix is represented by a

symbol (1 or 0) plotted according to that author's fraction of row votes (horizontally) and fraction of column votes (vertically). The plotting symbol for

the true author is '1' and the plotting symbol for all other authors is '0'.

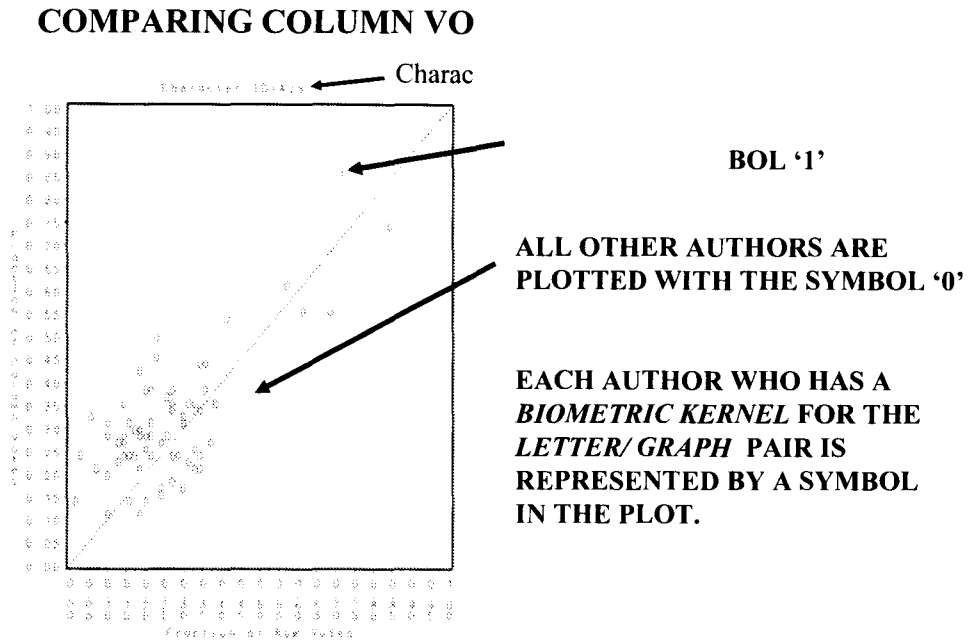


Figure 5: Comparing Row and Column Voting

We seek a single value (statistic) that reflects both how close the fraction of row votes is to 1 and how close the fraction of column votes is to 1. Note that

each symbol (author) in Figure 5 can be associated with a two-by-two frequency table. This table association is defined in Figure 6.

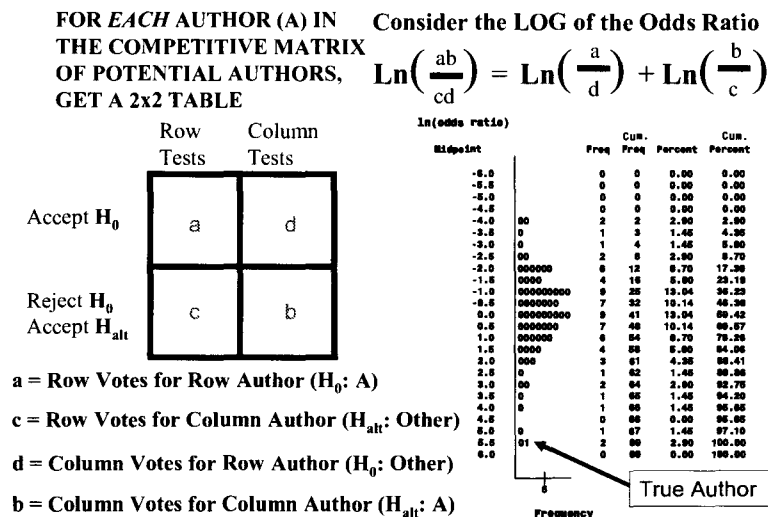


Figure 6: The Log of the Odds Ratio

4.2 The Log of the Odds Ratio

Considering this two-by-two table, a single value (statistic) with the desired properties is the log of the odds ratio, that is, $\log(ab/cd)$. A histogram for $\log(ab/cd)$ for the data displayed in the plot of Figure 5 is presented in Figure 6. Note that the true author (symbol '1') and another author (symbol '0') appear tied for the highest value of $\log(ab/cd)$. These two authors are out in the far tail of the distribution of $\log(ab/cd)$ values for all of the 69 authors who are in the Competitive Matrix.

Figure 6 parses the $\log(ab/cd)$ value into the sum of two log odds values:

$$\log(ab/cd) = \log(a/d) + \log(b/c).$$

This breakdown of the log of the odds ratio helps us to appreciate the power for true author identification in this quantity. Figure 7 plots a versus d for the true authors associated with all characters in the held out two London letters for all authors in our database. Note in Figure 7 the strong pattern of plotted points with $a > d$; such points will give large values of $\log(a/d)$.

Alternatively, Figure 8 plots a versus d for the authors who are not the true authors for the same

characters in the held out two London letters. Note in Figure 8, the symmetric pattern of plotted points around the 45 degree line where $a = d$; for such points the values of $\log(a/d)$ will be symmetric around 0.

Figure 9 plots c versus b for the true authors associated with all characters in the held out two London letters for all authors in our database. Note in Figure 9, the strong pattern of plotted points with $b > c$; such points will give large values of $\log(b/c)$.

Alternatively, Figure 10 plots c versus b for the authors who are not the true authors for the same characters in the held out two London letters. Note in Figure 10, the symmetric pattern of plotted points around the 45 degree line where $b = c$; for such points the values of $\log(b/c)$ will be symmetric around 0.

Figure 11 presents a histogram of $\log(ab/cd)$ for all authors (both true authors and others) from the Competitive Matrices for each character in all held out London letters. Note the smooth, symmetric structure of the histogram. The parts of the histogram associated with values for true authors are darkened in; note that the values for true authors are predominantly positive whereas the values for other authors are randomly positive or negative.

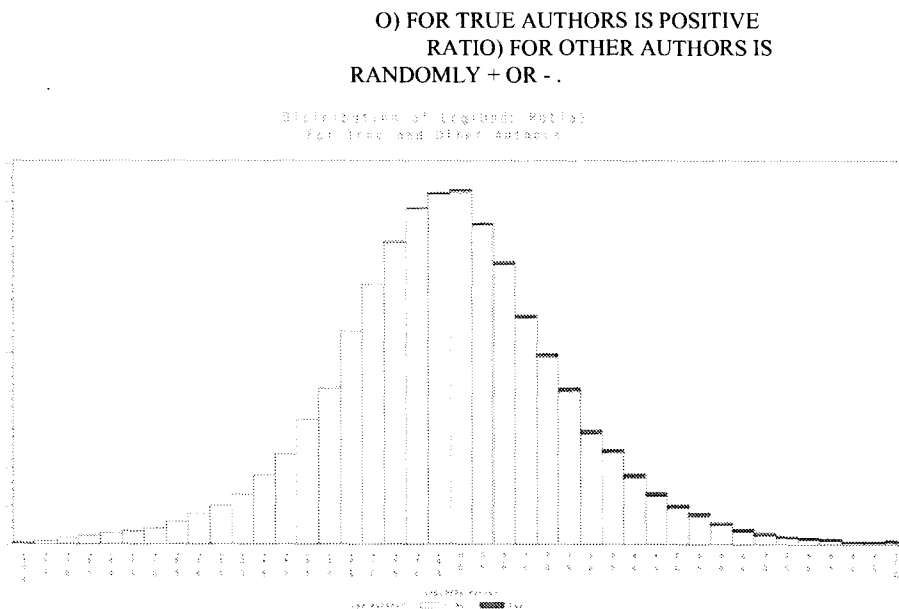
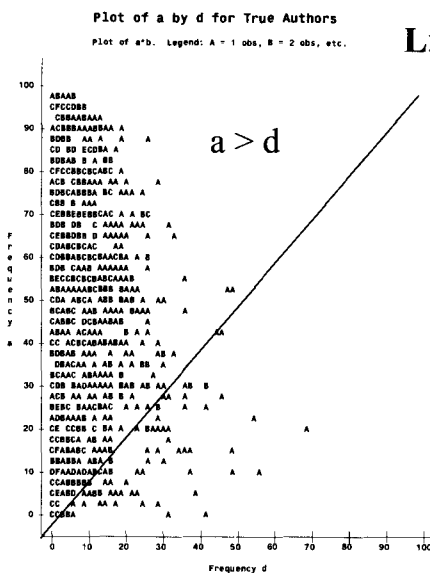


Figure 11: The distribution of the Log of the odds ratio



$$\ln\left(\frac{ab}{cd}\right) = \ln\left(\frac{a}{d}\right) + \ln\left(\frac{b}{c}\right)$$

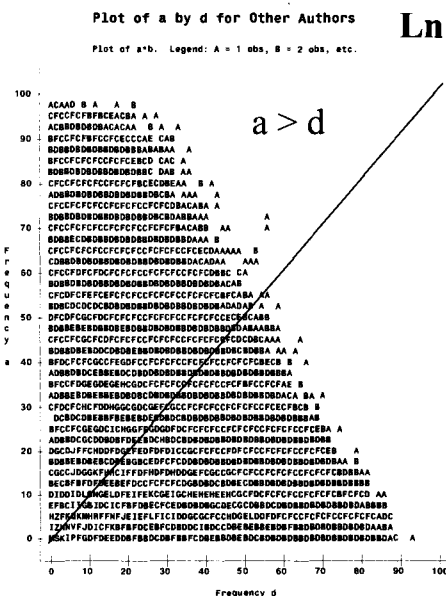
Row Tests	Column Tests
a	d
c	b

Accept H_0

Reject H_0
Accept H_{alt}

$a/d =$ Odds that an Accepted H_0 is from a Row Test

Figure 7: Plot of a versus d for True Authors



$$\ln\left(\frac{ab}{cd}\right) = \ln\left(\frac{a}{d}\right) + \ln\left(\frac{b}{c}\right)$$

Row Tests	Column Tests
a	d
c	b

Accept H_0

Reject H_0
Accept H_{alt}

$a/d =$ Odds that an Accepted H_0 is from a Row Test

Figure 8: Plot of a versus d for Other Authors

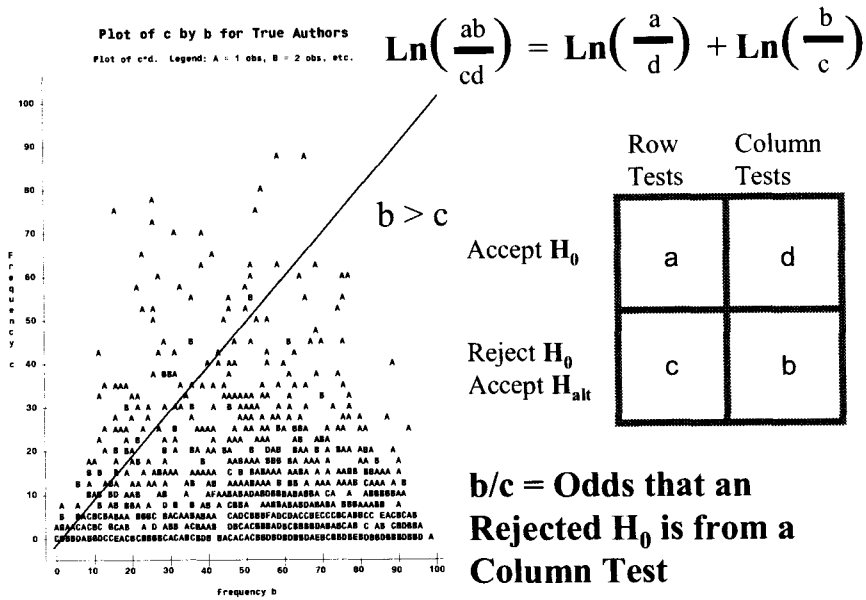


Figure 9: Plot of b versus c for True Authors

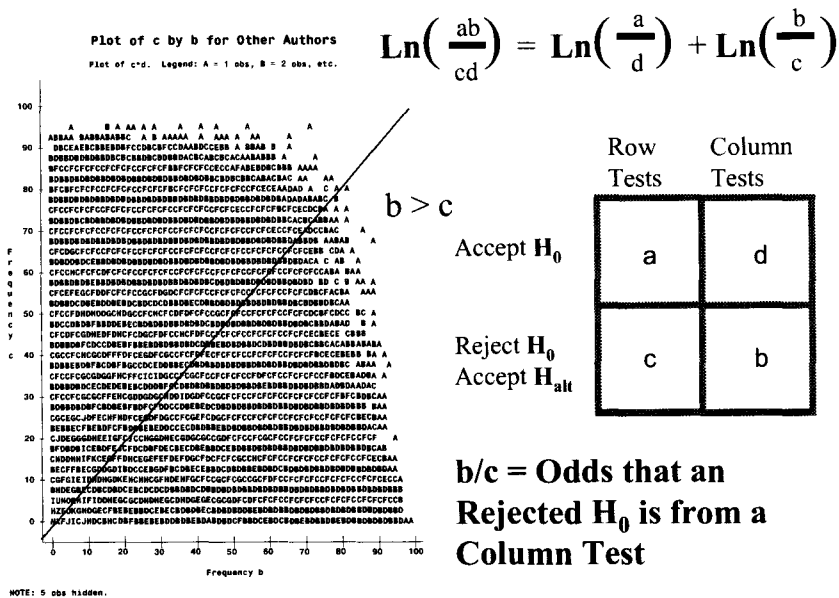


Figure 10: Plot of b versus c for Other Authors

4.3 Testing Multiple Characters

If we take multiple characters from a single author's held out London letters, then the sum of the $\log(ab/cd)$ values for these letters assesses the aggregate likelihood of authorship for the multiple characters. A true author will have a preponderance

of positive $\log(ab/cd)$ values which will lead to a strongly positive sum of $\log(ab/cd)$ values. This should discriminate the true author from other authors in the database. Figure 12 demonstrates this discrimination for a particular author based on fifteen randomly selected characters from that author's held out London letters.

FOR A PARTICULAR AUTHOR AMONG A GROUP OF 100 AUTHORS,

- TAKE 15 CHARACTERS AT RANDOM FROM THE HELD OUT PARAGRAPHS.**
- TEST EACH OF THE 15 CHARACTERS**
- SUM THE TEST CHARACTER LN(ODDS RATIO) VALUES FOR EACH OF THE 100 AUTHORS**

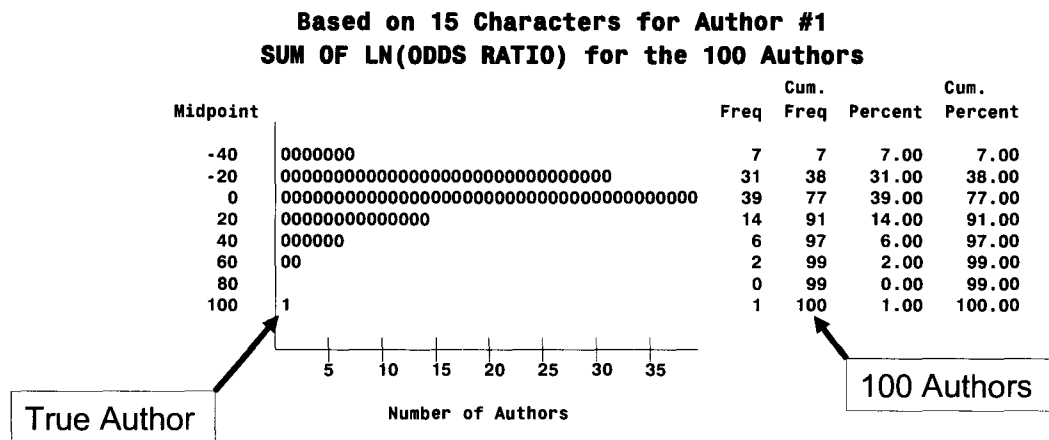


Figure 12: Sum of Logs of Odds Ratios for 15 characters

To investigate the accuracy of author identification based on the log of the odds ratio, we took random samples of characters from each author's held out London letters and hypothetically considered these characters to be from a note written by an unknown author in our modeling database. We tested all of these characters and selected the author with the largest sum of logs of the odds ratio to be our candidate for the true author. We repeated this random sampling 30 times for each sample size

(number of characters) from 5 to 30. The two graphs in Figure 13 summarize the accuracy results. The first graph presents the average percent of samples for which the true author was in first place, second place, etc., when authors were ranked by decreasing sums of logs of the odds ratio. The second graph presents the number of samples for which 96 out of 100, 97 out of 100, etc., true authors had the largest sum of the logs of the odds ratio thereby selecting them as our candidate for true author.

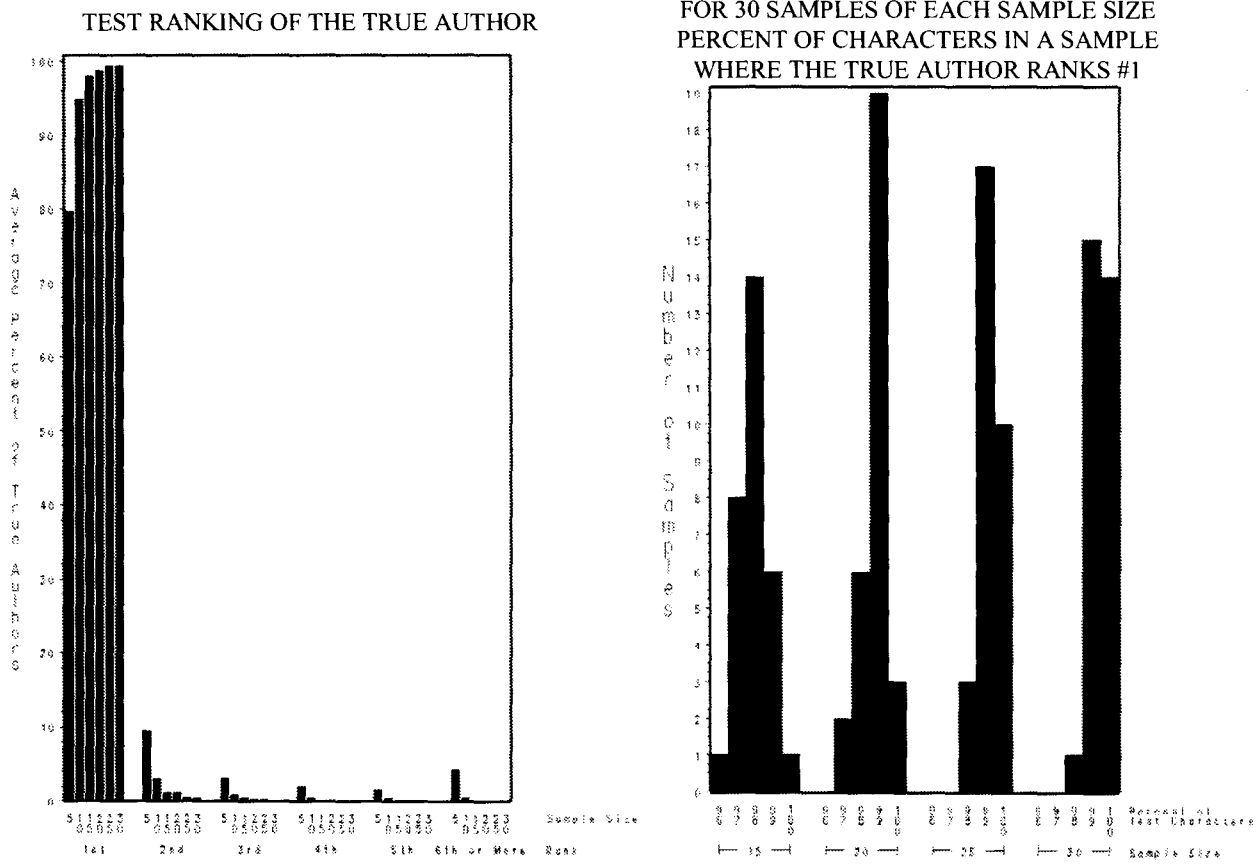


Figure 13: The accuracy of the Biometric Identification

5 Conclusion

We have demonstrated that the quantification of handwriting by the assignment of a graph to each character provides a very powerful basis for biometric identification. The concepts of Biometric Kernels and Competitive Matrices are used to build a biometric identification database for use in determining the identity of an unknown author of a writing sample.

We are applying the technology described in this article to applications in the areas of Individual Identification, Document Clustering and Forensic Document Examination.

Tradeoff Studies about Storage and Retrieval Efficiency of Boundary Data Representations for LLS, TIGER and DLG Data Structures

David Clutter Peter Bajcsy
National Center for Supercomputing
Applications (NCSA), University of Illinois at
Urbana-Champaign (UIUC)
605 East Springfield Avenue, Champaign, IL
61820

Abstract

We present our theoretical comparisons and experimental evaluations of three boundary data representations in terms of storage and information retrieval efficiency. We focus on three boundary data representations, such as, location list data structure (LLS), digital line graphs (DLGs) and topologically integrated geographic encoding and referencing (TIGER) data organizations. These three boundary data representations are used frequently in the GIS domain, and are known as ESRI Shapefiles (LLS), the SSURGO DLG-3 soil files (DLG), and the U.S. Census Bureau 2000 TIGER/Line files (TIGER). Boundary information is viewed as an efficient representation of image documents describing spatial regions. The goal of our work is to study the impacts of choosing boundary information representation on document image management and information retrieval, as well as to improve our understanding of the processing noise introduced during representation conversions.

Our storage and retrieval efficiency tradeoff evaluations are based on load time, computer memory, and hard disk space requirements. The experimental measurements are obtained with test data sets derived from the SSURGO DLG-3 soil files and the U.S. Census Bureau 2000 TIGER/Line files. Based on our experiments, we concluded that LLS files will provide the fastest boundary retrieval (40 times faster than TIGER and 2.5 times faster than DLG) at the price of file size (storage redundancy for LLS files is between 70% and 180% in our experiments). DLG format offers a smaller file size, but is less efficient for boundary retrieval. TIGER format also offers a compact physical representation, at the cost of more processing for boundary retrievals. These findings provide quantitative support for institutional document image management decisions.

1 Introduction

Boundary information is viewed as an efficient representation of image documents describing spatial regions and is important for document image management and information retrieval. Boundary information represents one type of vector information [1, Chapter 15]. Boundaries (or contours or outlines) are mathematically described as convex or non-convex polygons. One boundary can be formed by a set of polygons, for instance, a donut shape boundary. Each polygon consists of an ordered set of points or vertices. In most GIS applications, points are georeferenced so that boundary information can be integrated with raster information. GIS examples of boundary information would be parcels, eco-regions, watersheds, soil regions, counties, Census tracts or U.S. postal zip codes. The goal of our work is to study the impacts of choosing boundary information representation on document image management and informational retrieval, as well as improve our understanding of the processing noise introduced during representation conversions.

In general, boundaries can be spatially related or can be spatially independent. The spatially related boundaries can be either partially overlapping or totally overlapping, such as, one contour being a subset of another set of boundaries. For example, watershed and U.S. postal zip codes boundaries are spatially independent while the U.S. Census Bureau tracts and blocks are spatially dependent in such a way that every tract is formed by a set of blocks.

Given the variety of boundary information, researchers have developed numerous file formats for storing boundary information. These file formats are designated in general for storing any vector data. Vector data contain points, lines, arcs, polygons or any combinations of these elements. Any vector data element can be represented in a reference domain defined by a latitude/longitude, UTM or pixel

coordinate system. The challenge in storing vector data is to organize the data such that the positions and geographic meanings of vector data elements are efficiently stored and easily extracted.

Among all vector data representations in files, the following data structures have been used frequently: location list data structure (LLS), point dictionary structure (PDS), dual independent map encoding structure (DIME), chain file structure (CFS), digital line graphs (DLGs) and topologically integrated geographic encoding and referencing (TIGER) files. For detailed description of each data structure we refer a reader to [1].

The motivation of our work came from the fact that while boundary data types are preferred over raster data types when it comes to storing boundary information, there are multiple memory storage schemes for boundary information, as listed in the previous paragraph. However, choosing the storage scheme that minimizes memory requirements might have a detrimental impact on boundary information retrieval efficiency. Thus, our objective is to evaluate quantitatively the tradeoffs between storage and retrieval efficiency of multiple boundary data representations for LLS, TIGER and DLG data structures. The outcomes of our evaluations are useful for (a) institutional decisions about archiving and retrieving geospatial boundary information, and (b) custom applications that perform processing of large size, geospatial boundary data sets.

In this work, we evaluate three boundary data representations for efficient boundary information storage and retrieval. These three data representations include (1) Census 2000 TIGER/Line files defined by the U.S. Census Bureau and saved in topologically integrated geographic encoding and referencing (TIGER) data structures, (2) shapefiles defined by the Environmental Systems Research Institute (ESRI) and stored in location list data structure (LLS) data structures, and (3) SSURGO DLG-3 soil boundaries prepared by the United States Geological Survey (USGS) and stored in digital line graphs (DLGs) data structures. We overview the three data file formats first. Next, we present our experimental results, and pairwise analysis of experimental results. Finally, we summarize our work and add a few observations about other possible trade-off metrics that might be considered for making institutional decisions.

2 SSURGO DLG-3 Soil Files

The Soil Survey Geographic (SSURGO) Digital Line Graphs (DLG) files provide geographical information on the boundaries of soil types [9], [10], [11]. The SSURGO data sets provide the highest spatial resolution of soil type information among the three soil

geographic data bases, such as, the Soil Survey Geographic (SSURGO) data base, the State Soil Geographic (STATSGO) data base, and the National Soil Geographic (NATSGO) data base.

2.1 File Format Description

DLG File Structure: The DLG file structure is designed to support all categories of spatial data that can be represented on a map. Three distinct types of DLG are defined. Large-scale DLG data is digitized from 1:24,000-scale USGS topographic quadrangles (SSURGO). Intermediate-scale DLG data is digitized from 1:100,000-scale USGS quadrangles (STATSGO). Small-scale DLG data is digitized from 1:2,000,000-scale sectional maps (NATSGO). Furthermore, three levels of DLG data were defined in terms of the number of attributes. It was found that the widest user community would be served by DLG Level 3 (DLG-3) data, which allows for the highest resolution (SSURGO) and highest number of attributes to be encoded (Level 3). The lesser levels of DLG encoding are unused. DLG-3 encodes attributes using two codes: a major code and a minor code. Similar attributes share a major code. The SSURGO DLG-3 soil database uses both the major code and minor code to encode the primary key into a relational database to further describe an area.

We gathered the SSURGO DLG-3 files for a few counties in Illinois from <http://www.ncgc.nrcs.usda.gov/branch/ssb/products/ssurgo/data/index.html>. There are two files for each county, such as, `dlg.zip` (digital line graph or DLG) and `tab.zip` (ASCII attribute data available in Microsoft Access 97 or later template database). The files contain soil boundaries of 18,000 soil series recognized in the United States. For the integration purposes, we have explored the following information from the DLG-3 documentation: (a) file naming convention, (b) spatial resolution, (c) spatial accuracy, (d) geographic coordinate system and (e) storage format. In terms of file naming convention, the `dlg.zip` file would contain files with the following suffixes:

- af - soil polygon DLG-3 file,
- aa - soil polygon attribute file,
- sf - special soil point and line DLG-3 file, and
- sa - special soil point and line attribute file.

Regarding spatial resolution, soil survey is mapped at a scale ranging from 1:12,000 to 1:63,360. The SSURGO soil boundaries meet the accuracy standards for the USGS 7.5-minute topographic quadrangles or the 1:12,000 or 1:24,000 orthophotoquads. Finally, the storage format is Digital Line Graph optional format with the attribute table data archived in ASCII table or INFORMIX table format.

DLG Georeferencing Information: In terms of a geographic coordinate system, coordinates are derived from the North American Datum of 1983 reference

system that is based upon the Geodetic Reference System of 1980. DLG data are recorded in either the Universal Transverse Mercator (UTM) system or are projected using the Albers Equal-Area Conic projection. SSURGO DLG-3 data are normally reported in the UTM system. STATSGO DLG data are reported using the Albers Equal-Area Conic projection.

DLG Data Description: DLG data are reported as nodes, lines, and areas. Lines are composed of a series of nodes, and areas are composed of lists of lines (or optionally nodes). The composition of an area or a line can be encoded either as a list of the nodes that make up the element, or as a list of points. Due to this hierarchical structure, each element must be encoded with a unique identifier.

A node is a coordinate on a map. Each node has an Easting value and a Northing value in the UTM coordinate system. Nodes define the points of each line and are encoded with (1) a unique identifier and (2) the coordinates that the node represents. Nodes can also be encoded with attributes, if desired. Additionally, the DLG format specification allows for a list of all lines that begin and end at a node to be encoded in the record for a node. This is redundant information, however, for it is reflected in the line records as well.

Lines are a series of nodes. Each line is encoded with a unique identifier, as well as its starting node and ending node. The coordinates that a line follows are also listed. In addition, a line can be encoded with attributes.

An area is an enclosed section. Areas can be encoded as either a sequence of lines or a sequence of nodes. When encoded as a sequence of lines, the area will contain a list of the lines that the boundary of the area follows. This list contains the unique identifier for each line; negative values signify that the points in the line should be reversed. Islands within an area are delimited by a '0' in the list of lines. Areas are specified in a clockwise direction around the perimeter of the area, and islands are specified in a counter-clockwise direction. In addition, an area can be encoded with major and minor code pairs. When encoded as a sequence of nodes, the area will contain a list of the nodes make up the boundary of the area.

Software Development for SSURGO DLG-3 Files: First, we implemented a loader for SSURGO DLG-3 files and added it to the list of other GIS files supported by the NCSA I2K software package [5]. Next, we extended our 2D visualization to support visualization SSURGO DLG-3 files. We can visualize multiple georeferenced vector data structures (boundaries and sets of points) simultaneously. Third, we develop a conversion function from SSURGO DLG-3 data structure to ESRI Shapefile (LLS) data structure that was needed for tradeoff comparison purposes.

The details of boundary information retrieval from DLG-3 file format can be described as follows. The

DLG file format defines objects using a hierarchical structure. The lowest objects in the hierarchy must be retrieved prior to higher objects in the hierarchy. Thus, in order to retrieve an area, all lines that make up the area's boundary must be retrieved beforehand. Therefore, the DLG-3 loader in I2K will read all the defined lines first. The lines are kept in a lookup table, and indexed by their unique identifier for later use. The size of this structure is directly proportional to the number of lines.

Next, the areas are retrieved by populating I2K defined data structures for boundary information denoted a ShapeObject. In the ShapeObject, an area has a list of the coordinates that make up its boundary. This list is dynamically constructed when reading an area. Areas that share a boundary will have copies of the common coordinates. Once all areas have been read and processed, the lookup table containing the lines can be safely discarded. Finally, the coordinates for the areas are copied into a ShapeObject.

2.2 Theoretical Evaluation

Memory requirements: The DLG-3 optional format used in SSURGO soil databases provides a compact physical representation of the boundaries of soil types over a geographic area. There is little redundancy in a DLG-3 file. Each area is a list of lines that do not cross. The lines must share the same endpoints in order to fully define an area. Thus, the only redundant information is the endpoints of each line. The points of adjacent polygons will be specified only once; in a line, or series of lines. The boundary between adjacent, non-overlapping polygons is represented as the same series of line identifiers in the file. In addition, representing all data in a fixed-length ASCII form makes for smaller, highly compressible files. Abundant white space exists in DLG-3 files to maintain the fixed length. Typical compression algorithms will compress a series of identical characters efficiently. Thus, when a DLG-3 file is subject to compression, the white space will compress well.

Boundary information retrieval requirements: The boundary information retrieval from DLG-3 file format can require significant processing resources. All boundary coordinates are stored as ASCII characters in a DLG file. In order to use the polygons specified in a file, each coordinate must be converted into a native numeric value. This conversion can be quite costly, and takes approximately 27% of the time to load SSURGO DLG-3 files in I2K.

3 Census 2000 TIGER/Line Files

The Census 2000 TIGER/Line Files provide geographical information on the boundaries of counties, zip codes, voting districts, and a geographic hierarchy of census relevant territories, e.g., census tracts that are composed of block groups, which are in turn composed

of blocks. It also contains information on roads, rivers, landmarks, airports, etc, including both latitude/longitude coordinates and corresponding addresses [2]. A detailed digital map of the United States, including the ability to look up addresses, could therefore be created through processing of the TIGER/Line files.

3.1 File Format Description

Because the density of data in the TIGER/Line files comes at the price of a complex encoding, extracting all available information from TIGER/Line files is a major task. In this work, our focus is primarily on extracting boundary information of regions and hence other available information in TIGER/Line files is not described here.

TIGER/Line files are based on an elaboration of the chain file structure (CFS) [1], where the primary element of information is an edge. Each edge has a unique ID number (TIGER/Line ID or TLID) and is defined by two end points. In addition, each edge then has polygons associated with its left and right sides, which in turn are associated with a county, zip code, census tract, etc. The edge is also associated with a set of shape points, which provide the actual form an edge takes. The use of shape points allows for fewer polygons to be stored.

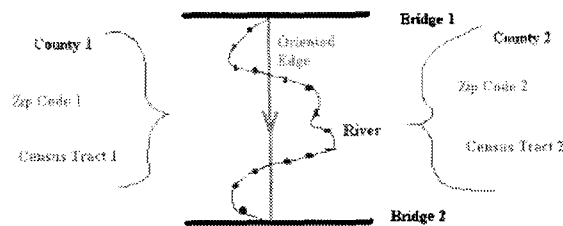


Figure 1: Illustration of the role of shape points.

To illustrate the role of shape points, imagine a winding river that is crossed by two bridges a mile apart, and that the river is a county boundary and therefore of interest to the user (see Figure 1). The erratic path of the river requires many points to define it, but the regions on either side of it do not change from one point to the next, only when the next bridge is reached. In this case, the two bridge/river intersections would be the end points of an edge and the exact path of the river would be represented as shape points. As a result, only one set of polygons (one on either side of the river) is necessary to represent the boundary information of many small, shape defining edges of a boundary.

This kind of vector representation has significant advantages over other methods in terms of storage space. To illustrate this point, consider that many

boundaries will share the same border edges. These boundaries belong to not only neighboring regions of the same type, but also to different kinds of regions in the geographic hierarchy. As a result, storing the data contained in the TIGER/Line files in a basic location list data structure (LLS) such as ESRI Shapefiles, where every boundary stores its own latitude/longitude point, would introduce a significant amount of redundancy to an already restrictively large data set.

In contrast to its apparent storage efficiency, the TIGER vector data representation is very inefficient for boundary information retrieval and requires extensive processing. From a retrieval standpoint, an efficient representation would enable direct recovery of the entire boundary of a region as a list of consecutive points. The conversion between the memory efficient (concise) and retrieval efficient forms of the data is quite laborious in terms of both software development and computation time.

Another advantage of the TIGER/Line file representation is that each type of GIS information is self-contained in a subset of files. As a result users can process only the desired information by loading a selected subset of relevant files. For example, each primary region (county) is fully represented by a maximum of 17 files. Therefore, the landmark information is separate from the county boundary definition information, which is separate from the street address information, etc. Those files that are relevant to the boundary point extraction, and the attributes of those files that are of interest, are the following:

- Record Type 1: Edge ID (TLID), Lat/Long of End Points
 - Record Type 2: TLID, Shape Points
 - Record Type I: TLID, Polygon ID Left, Polygon ID Right
 - Record Type S: Polygon ID, Zip Code, County, Census Tract, Block Group, etc.
 - Record Type P: Polygon ID, Internal Point (Lat/Long).
- We denote this subset of files as “Census boundary records”.

3.2 Theoretical Evaluations

This work extends our previous study about the tradeoffs between U.S. Census Bureau TIGER and ESRI Shapefile data representations that are documented in [7].

4 ESRI Shapefiles

A shapefile is a special data file format that stores non-topological geometry and attribute information for the spatial features in a data set. The geometry for a feature

is stored as a shape comprising a set of vector coordinates in a location list data structure (LLS). Shapefiles can support point, line, and area features. Area features are represented as closed loop polygons.

4.1 File Format Description

A shapefile must strictly conform to the ESRI specifications [4]. It consists of a main file, an index file, and a dBASE table. The **main file** is a direct access, variable-record-length file in which each record describes a shape with a list of its vertices. In the **index file**, each record contains the offset of the corresponding main file record from the beginning of the main file. The **dBASE table** contains feature attributes with one record per feature. The one-to-one relationship between geometry and attributes is based on record number. Attribute records in the dBASE file must be in the same order as records in the main file.

All file names adhere to the ESRI Shapefile 8.3 naming convention. The 8.3 naming convention restricts the name of a file to a maximum of 8 characters, followed by a 3 letter file extension. The main file, the index file, and the dBASE file have the same prefix. The suffix for the main file is ".shp". The suffix for the index file is ".shx". The suffix for the dBASE table is ".dbf".

Examples:

1. main file: counties.shp
2. index file: counties.shx
3. dBASE table: counties.dbf

The implementation of shapefile loading, writing and visualization routines was straightforward since the I2K ShapeObject data structure maps directly to the shapefile file organization.

4.2 Theoretical Evaluation

There are numerous reasons for using ESRI Shapefiles. ESRI Shapefiles do not have the processing overhead of a topological data structure such as a TIGER file. They have advantages over other data sources, such as faster drawing speed and edit ability. ESRI Shapefiles handle single features that overlap or are noncontiguous. They also typically require more disk space but are easier to read and write. However, the drawbacks of ESRI Shapefiles are in their storage inefficiency and poor scalability. We will quantify these tradeoffs in the experimental section.

5 Experimental Evaluations

In this section, our goals are (a) to experimentally evaluate the tradeoffs between storage and retrieval efficiency, and (b) to explain the tradeoffs by comparing fundamental format differences. In order to perform experimental tradeoff evaluations, we used two

datasets including (1) the SSURGO soil boundaries for Madison County, IL, stored in DLG-3 file format and (2) the U.S. Census Bureau boundaries of Illinois counties, zip codes, census block and census tracts stored in TIGER/Line file format. The preparation of these two data sets is outlined in Section 5.1. The results of all experiments are provided in Sections 5.2 and include comparisons of DLG & LLS, and DLG & TIGER & LLS. Sections 5.3, 5.4 and 5.5 explain the pair-wise format comparisons based on the experimental results.

5.1 Data Preparation

It is apparent that the experimental evaluations will depend on the size of test data. Ideally, one would like to show results as a function of input file size. However, the practical difficulty arises when one is looking for those test data sets that contain identical boundary information but are represented by LLS, TIGER and DLG files. We were not able to find such files.

We explored the possibility of finding software tools that would convert vector files from one file format to another so that we could create multiple test files with identical boundary information stored in LLS, TIGER and DLG formats. We have concluded that while LLS formats (ESRI Shapefiles) are supported by most GIS software packages, there is a very limited support for DLG and TIGER file formats. This corresponds to our assessment of the implementation complexity to support loading of TIGER, DLG and LLS formats in this order from the most time consuming to the least time consuming. The implementation effort usually doubles when both loading and writing routines have to be supported.

Based on our findings about conversion tools and the availability of GIS software packages at our institution, we created data sets by (1) implementing TIGER to LLS, and DLG to LLS conversions, and (2) using ArcToolBox for LLS to DLG conversion. We created several test data sets that are described next.

In the first experimental tradeoff evaluation, we used a file pair consisting of the original DLG file (SSURGO soil boundaries) and the LLS file converted using I2K. This file pair is denoted as the test data set #1.

In the second experimental tradeoff evaluation, we prepared a triplet of files consisting of (a) the original TIGER files for the state of Illinois, (b) the LLS files obtained by extracting the U.S. Census Bureau boundaries of counties, zip codes, census block and census tracts from the TIGER files and converting them by using our software implementation, and (c) the DLG file converted from the already obtained LLS file using ArcToolBox. This triplet of files provides a test data set for fair performance evaluations in terms of "Total Load Time" and Load RAM Required" parameters. However, this test data set cannot be used for

Table 1: Test data#1: SSURGO Soil Database, Madison County, IL. Loading time includes all SSURGO soil boundaries. Hard disk measurements pertain to all boundaries in the original SSURGO files.

	Total Load Time (s)		Load RAM Required (MB)	Hard Disk (MB)		Number of Nodes
	Zip	Unzip		Zip	Unzip	
LLS (Shapefile)		41.36	290	65	90	2,787,490
DLG	105.72	103.72	380	23	79	2,787,790

Table 2: Test data#2: U.S. Census Bureau 2000 TIGER/Line files for the state of Illinois (102 counties). Loading is constrained to block groups, zcta, census tract, and counties ((Total Load Time and Load RAM Required parameters). Hard Disk and Number of Nodes measurements for LLS and DLG formats contain only block groups, zcta, census tract, and county boundaries, whereas the same measurements for TIGER format include all types of boundary information for the state of Illinois.

	Total Load Time (s)	Load RAM Required (MB)	Hard Disk (MB)		Number of Nodes
	Unzip		Zip	Unzip	
TIGER	1300.2	200	112	940	2,176,719
LLS	12.7	37	27	47	641,955
DLG-3	12.9	52	8	24	457,850

performance evaluations in terms of “Hard Disk” because the TIGER files include all boundary types (including voting districts, and so on), of which four were extracted to LLS and DLG file formats. This file triplet is denoted as the test data set #2.

We expanded the second experimental tradeoff evaluations in Section 5.2 by partitioning the test data set #2. We used sub-sets of the original TIGER files for the state of Illinois in order to vary the number of nodes. In order to explore load time dependency on the number of nodes (boundary points), we selected 1, 2, 3, 4, 10, 15, or 24 counties from the original TIGER files, and formed several triplets of test data sets (TIGER, LLS and DLG). We always chose a subset of counties forming geographically contiguous regions so that neighboring counties would have some overlap of boundary points. This set of file triplets is denoted as the test data set #3.

5.2 TIGER, LLS, and DLG Tradeoff Evaluations

The experimental results of our tradeoff evaluations between storage and retrieval efficiency are presented in Tables 1 and 2. As described in the previous section, the test data sets #1 and #2 {(DLG, LLS) and (TIGER, LLS, DLG)} were formed from the original DLG and TIGER files by converting them into other file formats

using Arc ToolBox and our software. Each file format was then read in separately, and the storage and loading measurements were recorded in Tables 1 and 2.

Before explaining the experimental results by comparing pairs of file formats presented in Sections 2, 3 and 4, we posed the following two questions. First, is there any dependency of storage on the boundary content? In other words, if we had a file with watershed and zip code boundaries, would the results be different from evaluating Census tracts and blocks, and how? Second, can we predict the total load time as a function of the number of polygons/nodes without exhaustive experimentation? Or in other words, what would be the dependency between boundary information retrieval and the number of retrieved nodes?

Storage Dependency on Boundary Content: The answer to the first question is related to the amount of boundary overlap. Ideally, one would experiment with sets of boundaries that span cases from a zero overlap (e.g., non-adjacent county boundaries) to an overlapping hierarchy of polygons (census blocks, block groups and tracts). Our data sets represent the cases of partial overlap (SSURGO) and large overlap (TIGER) of boundaries. Thus, the experimental results will vary as a function of boundary content in the following way: the more overlapping boundaries, the smaller hard disk requirements for TIGER format in

comparison with DLG and LLS (in this order), and the smaller load RAM requirements for LLS format in comparison with DLG and TIGER.

Our conclusion is supported by comparing the number of loaded nodes versus the number of unique nodes using the test data sets #1 and #3, and by inspecting the LLS files. By evaluating the ratio s of these two numbers (loaded nodes versus unique nodes) using the test data #2 (partial boundary overlap), we obtain s equal to 2.02 (5630800/2787490). The same evaluation of the ratio s using the data set #3 (large boundary overlap) led to an average ratio value equal to 2.6416. The measurements using the test data set #3 (ZCTA, Block Group (BG), Census Tract (CT), and County boundaries for 1, 2, 3, 4, 10, 15, and 24 Illinois counties) are shown in Figure 2.

We took additional measurements to compute the ratio s for (a) watershed and county boundaries ($s = 84,601/47,636=1.776$), and (b) watershed and ZCTAs boundaries ($s=344,533/201,767=1.708$). We observed that approximately 70% of the points in both (a) and (b) are shared between multiple boundaries. Thus, the inefficiency of LLS format due to the duplicate points of neighboring boundaries would not decrease below $s=1.7$ for the test data.

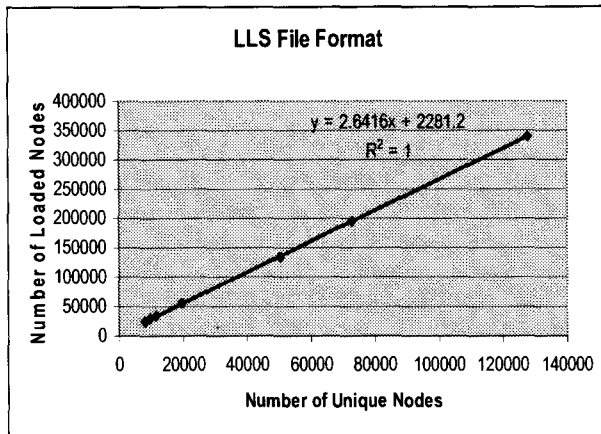


Figure 2: Storage efficiency measurements of LLS files using the test data set #3 (Hierarchical boundary content). The points correspond to evaluations for data sets with boundaries for 1, 2, 3, 4, 10, 15, and 24 Illinois counties.

Boundary Information Retrieval Dependency on Number of Nodes: In order to answer the second question about the relationship between a load time and a number of nodes, we divided the Total Load Time into four components: t_1 , t_2 , t_3 and t_4 (see Equation below and Figure 3). The first component t_1 corresponds to the time to construct polygons from an ordered list of edges. The second component t_2 is for the time to create an ordered list of edges from an unordered set of edges. The third component t_3 represents the time to convert ASCII characters to numeric type values. The last component t_4 is the time

to load any sequence of bytes (ASCII characters or binary values) from a file. We introduce these time components based on our understanding of the three vector file formats.

$$Total\ Load\ Time = t_1 + t_2 + t_3 + t_4 \quad (1)$$

The zero and non-zero time components are summarized for each file format in Table 3. The total load time as a function of the number of nodes can be predicted by knowing that the time components t_1 , t_2 , t_3 and t_4 are linear with the increasing number of nodes. The quadratic dependency of the time component t_2 (creation of ordered list of edges) as a function of the increasing number of nodes is avoided by the fact that the unordered edges are grouped by counties rather than by states. Based on our empirical observations, $t_1 < t_2 < t_3$ for a fixed number of nodes, which leads to superior total load time for LLS format in comparison with DLG and TIGER formats (in this order). Our theoretical predicted Total Load Time as a function of the number of nodes is shown in Figure 3 and is independent of test data sets (addressed as the question number 1 above).

Table 3: Total Load Time decomposition.

Total Load Time=Sum(t_i)	t_1	t_2	t_3	t_4
LLS	X	0	0	X
DLG	X	0	X	X
TIGER	X	X	X	X

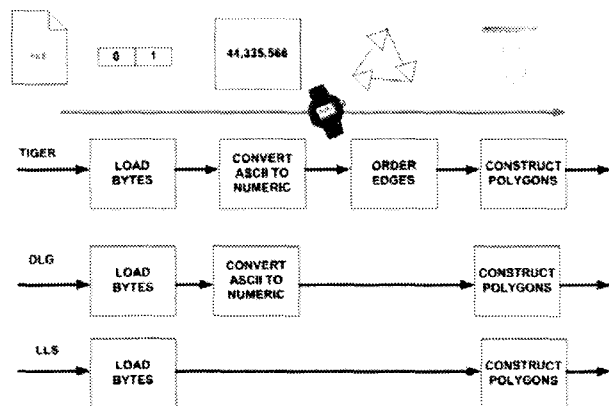


Figure 3: Total Load Time decomposition for TIGER, DLG and LLS file formats.

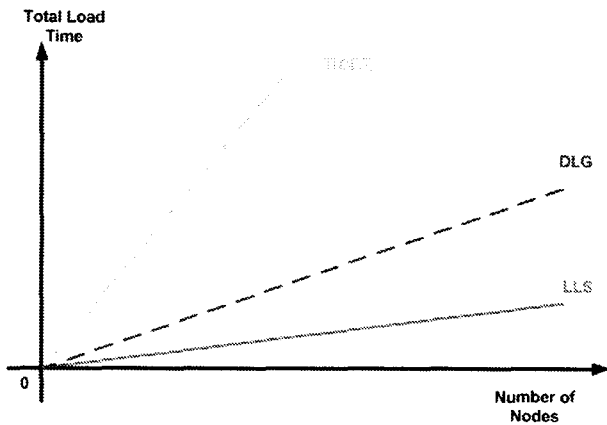


Figure 4 : Theoretically predicted Total Load Time as a function of the number of nodes.

We have obtained experimental measurements that support our theoretically predicted Total Load Time dependency on the number of nodes using the test data set #3. Figure 5 shows our measurements and linear trends, where the points correspond to data sets with boundaries for 1, 2, 3, 4, 10, 15, and 24 Illinois counties. These supporting measurements for “Total Load Time” and “Load RAM Required” were calculated by averaging three runs to load the ZCTA, Block Group (BG), Census Tract (CT), and County boundaries for each data set. The total number of nodes and the number of unique nodes were measured (a) by counting nodes inside of our software developed for loading LLS and DLG files, and (b) by summing end points and shape points for TIGER files according to the accompanying TIGER documentation. While TIGER files do not contain any duplicate points, LLS duplicate points were found using a hash table in our software.

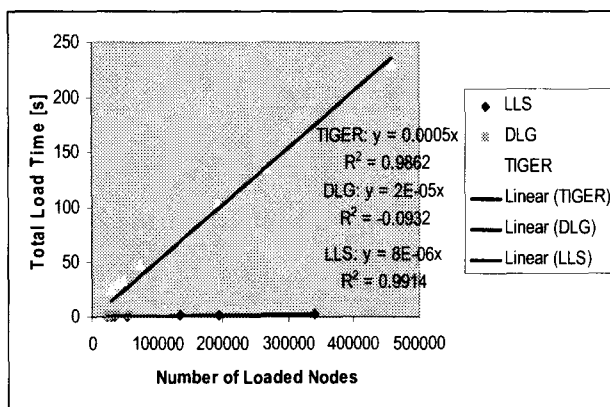


Figure 5 : Total Load Time vs. Number of Nodes for 1, 2, 3, 4, 10, 15, and 24 counties with a best-fit line.

According to Figure 5 and based on our test data set #3, the total loading time for TIGER files is approximately 40 times slower than for LLS files, and

the total loading time for DLG files is about 2.5 times slower than for LLS files. We collected measurements for only 1, 2, 3 and 4 county aggregations in the case of DLG format because the data preparation is very time consuming.

5.3 DLG and LLS Comparisons

The DLG optional and LLS (or ESRI Shapefile) formats specify boundaries over an area. Both formats have geographic information that allows the boundaries to be geo-referenced with other data sources. The formats differ in how the data is structured and stored.

The first primary difference between DLG and LLS is that DLG is stored in an ASCII format, while LLS is stored in a binary format. DLG files are comprised of ASCII characters organized into fixed-length logical records of 80 characters. When loading a DLG file, all data contained within must be converted to native data types. For example, a coordinate is stored as the ASCII characters “4598829.0” in the file. This must be read in and converted to its numeric value. ESRI Shapefile, on the other hand, stores the data as a series of bytes that can be quickly converted to a data type. For the previous example, the value “4598829.0” would be stored as 8 bytes that can be directly converted into a numeric value. However, it may be necessary to reverse the order of the bytes to account for the byte order (little or big endian). The reading (and possible reversing) of bytes for a shapefile is far simpler than the ASCII-to-native transformation needed for DLG.

This primary difference in representation (ASCII vs. binary) greatly affects the loading times of the two approaches. Each entry in a DLG soil database must be read individually, and then converted to a numeric value. This is the most time-consuming operation when loading the data, typically over 25% of the loading time of a DLG file. Loading ESRI Shapefile, however, is much quicker. It is simply reading a series of bytes from a file, with little conversion needed. This quickness comes at the price of a larger file size for the ESRI Shapefile. In an examination of one county, the DLG data needs approximately 79 MB of disk space uncompressed, 23 MB compressed. The ESRI Shapefile, on the other hand, needs 90 MB of disk space when uncompressed, and 65 MB when compressed. These results are summarized in Table 1 and Table 2. The difference in compressed sizes between the two encodings is attributable to their physical representations. DLG data contains fixed-length records with white space between elements to maintain the fixed length. This white space is insignificant and can be easily compressed. On the other hand, all binary data in an ESRI Shapefile are significant and cannot be easily compressed.

The second difference between DLG and LLS is the way how the data in a file are structured. DLG format uses nodes, lines, and areas to define its polygons. In

each of the SSURGO DLG datasets examined so far, nodes have not been used to define lines or areas. The lines are a series of coordinate values, and the areas have a list of the lines that make up the area. On the other hand, LLS format lists the bounding box and the points for each boundary contained within it. DLG format makes more efficient usage of space; areas that share lines will both reference the same line, while in a shapefile, each coordinate, including coordinates shared between different boundaries, is explicitly listed. In addition, this difference makes it necessary to first read all the lines in a DLG file before reading in the areas, because the areas are made up of a list of the lines. The lines have to be kept in a lookup table, and areas cannot be fully processed until all lines have been read.

The consequence of the second difference between DLG and LLS is that different data structures have to be used when loading these files. Our goal is to have one ShapeObject that contains all the polygons in a soil database. DLG format gives no hint as to how many points will be needed to store all the polygons in the DLG file. Furthermore, it does not give the bounding box for each polygon. In contrast, ESRI Shapefile stores these values so that it is possible (a) to pre-compute the space requirements needed and (b) to allocate arrays to hold the data when loading a Shapefile. With DLG, however, it would only be possible to pre-compute the sizes by reading in all data files twice. One time to determine the sizes, and one time to actually read in the data. In addition, the bounding box for each polygon is not stored in DLG, and must be found while reading in the coordinates of each area. This requires comparisons for each coordinate to find the bounding box. In our implementation, expandable arrays (or vectors) were used so that the files only had to be read in once. Then, once fully read, the data are copied into an array in the ShapeObject, of the exact size needed. The problem with this approach is that when the copy is made, two arrays must exist in memory. The first will be the array that contains the vector data. The second will be the new ShapeObject array to copy the contents of the vector into. This causes the memory requirements of DLG-3 files to balloon to twice the total necessary size in the worst case, when copying all the individual points of all the polygons into one ShapeObject.

The third difference between DLG and LLS is related to georeferencing information. SSURGO DLG files are stored as quarter-quadrangles. Each quadrangle represents 7.5 minutes of a degree of longitude and latitude. It is necessary to load 64 individual files to represent a one degree block. ESRI Shapefile does not need to be represented this way. However, Shapefiles could be stored in this way, if desired. All coordinates in SSURGO DLG files are stored in UTM format. This causes problems when geo-referencing the boundaries in I2K because the state of Illinois is located in both UTM zone 15 and UTM

zone 16. The solution was to immediately translate the UTM coordinates to latitude and longitude. Over 29% of the time to load a SSURGO DLG file was spent in the conversion from UTM coordinates to latitude and longitude. Each DLG file contains the UTM zone in the header information. ESRI Shapefile normally contains latitude and longitudinal geo-referencing information. No conversion was required when loading the shapefile in I2K. A potential drawback of the ESRI Shapefile format is that there is not a standard way to define the projection used in for the coordinates. DLG has a value in the header to signify if UTM or Albers projection is used. Also, some of the projection parameters are stored in the header of a DLG. Shapefiles, on the other hand, do not store projection information. This information could be stored with the meta data for a shapefile, but it is not required. This makes it difficult to distribute shapefiles with geo-referencing information other than standard latitude and longitude.

5.4 DLG and TIGER Comparisons

DLG and TIGER offer similar methods to encode vector data. TIGER's use of an edge with shape points corresponds directly to DLG's use of lines and coordinates. Likewise, a TIGER polygon is comprised of a series of edges, and a DLG area is made up of a series of lines. This provides a compact, human-readable representation of the vector data.

The two formats differ in the type of data that are encoded. DLG format typically encodes one layer of data in a file, such as the soil types used by SSURGO. Other layers, such as water boundaries, are encoded in separate files. This scheme introduces some redundancy between the layers. Layers are unrelated to one another, and any shared boundaries will be specified in each layer. For example, a soil layer encoded as a DLG may have boundaries defined along a river. A layer containing bodies of water may share the same boundaries, but the points will be specified again because the soil layer is unrelated to the body of water layer in DLG. TIGER format, on the other hand, groups all edges together, regardless of layer. The different metadata files are used to determine which edges to use. This format allows for less redundancy.

Polygons are retrieved very differently by the DLG and TIGER loaders. DLG format specifies the exact boundaries for each polygon. A list of lines defines the exact border of a polygon, and the lines are in the proper sequence. Since the lines appear in the proper sequence, the polygon can be quickly constructed after all line retrieval. In contrary to DLG format, the boundaries stored in TIGER format must be found programmatically. Each edge is labeled with the polygons that appear on the left and right of the edge. To construct a polygon A, you must first find all edges that border the polygon A. The edges only define the

end points of each edge, and not the order in which the edges should be connected. So the boundary of polygon A must be constructed programmatically by comparing the end points of each edge. Thus, the TIGER polygon construction is far more complex and time-consuming than the DLG polygon construction.

5.5 TIGER and LLS Comparisons

One can derive TIGER and LLS comparisons from the description provided in Sections 2, 3 and 4 that compare DLG and LLS, and TIGER and DLG formats. Since the experimental tradeoff evaluations of TIGER and LLS are summarized in Table 2, we devoted this section to the implementation of TIGER to LLS conversion.

The underlying principle of the conversion process from TIGER/Line files to ESRI Shapefiles could be compared to sorting points according to the order of boundary edges. This is illustrated in Figure 4. In reality, the conversion process begins by loading the raw TIGER/Line files into 2-D table-like data structures by making use of manually developed meta data files. Since the TIGER/Line files are fixed-width encoded flat files, meta data is necessary to define the indices of the first and last characters for each attribute in the lines of the flat file. This information, the attributes' names, and their type (integer, floating point number, string, etc) come from meta data files provided by the Census Bureau. The final piece of information contained in the meta data file is a "Remove Column" field, which dictates whether or not the attribute will be dropped from the table as it is read in. Attributes that are not used during the processing are removed early on for the sake of memory efficiency. The meta information for each Record Type is stored in a comma-separated-value (csv) file, which can easily be parsed into a table object, then accessed in that form by the routine that parses the main data file.

Once the TIGER/Line data are in the form of tables, they are streamed through a complex system of procedures, including conversion to several intermediate data structures, before being inserted into Hierarchical Boundary Objects (HBoundary) [7]. Each HBoundary represents one type of region (county, census tract, etc) for a single state. It can also be viewed as one master list of boundary points that all boundaries reference by pointers. The RAM memory savings of HBoundary versus ShapeObject for each point that is shared by two counties, two census tracts, and two block group boundaries is 30 bytes. For the state of Illinois, this optimization translated into a 38% reduction in memory usage (16.45 MB versus 26.64 MB).

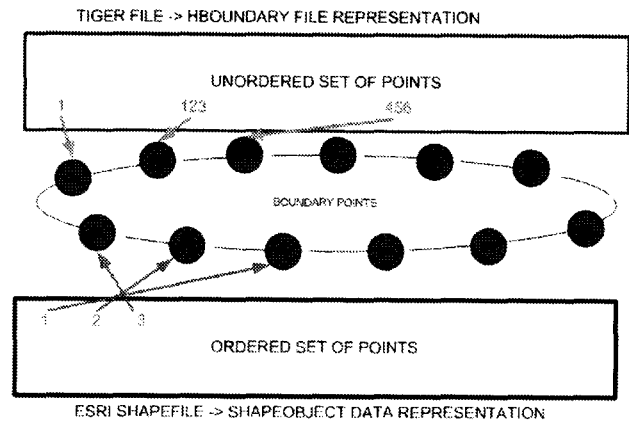


Figure 6: The TIGER/Line to ESRI Shapefiles conversion of boundary representation can be viewed as a transformation from an unordered set of points to a clock-wise ordered set of points.

Finally, the HBoundary object is converted into LLS format by constructing all polygons. The resulting LLS format file was tested by loading it into the commercial ArcExplorer software package [3]. For our experimental tradeoff evaluations, we extracted only a selected subset of Census boundary records from the Census 2000 TIGER/Line files. Thus, it is hard to evaluate loading RAM requirements for TIGER and other two formats since the HBoundary object contains all hierarchical boundaries and their associated information, while the converted LLS file contains only four types of boundaries (counties, ZCTAs, blocks and tracts) and extracted information about region names, neighboring regions to each boundary, and an internal point of each region.

6 Summary

In this paper we have investigated the storage and retrieval efficiency tradeoffs between the ESRI Shapefile (LLS), DLG, and TIGER formats. LLS files will provide the fastest method for boundary retrieval (40 times faster than TIGER and 2.5 times faster than DLG). All boundaries are stored in a binary format for quick retrieval. This speed comes at the price of file size. Each boundary in a LLS file contains all the points that make up the boundary. This introduces storage redundancy (between 70% and 180% redundancy in our experiments) since boundaries can be shared between different polygons. Digital Line Graphs reduce the amount of redundant data. This reduction is tempered by the need for more retrieval processing per boundary. The TIGER format further reduces the amount of data. TIGER format is the most compact representation that comes at the cost of the highest boundary retrieval requirements. Detailed information about these results can be found in Reference [12].

Our goal was to evaluate numerically the trade-

offs between storage and boundary retrieval requirements for the three vector files. The measurements about “Total Load Time”, “Load RAM Required” and “Hard Disk” as a function of “Number of Loaded/Unique Nodes” were used as our metric to demonstrate the trade-offs. Our measurements support the existing knowledge about the choice of a file format depending on the data content that is mapped to boundary overlaps. However, there are other metrics that might affect institutional decisions as well, and were not included in this study. We could enumerate a few metrics, such as (1) a cost of storage media and RAM, (2) a cost of software development to support complex file formats, (3) a preservation of storage media, (4) an availability of software tools for ingesting and processing certain file formats, or (5) an open source implementation of software tools that would allow tracking discrepancies in file format interpretation (loading) and replication (writing). While we did not quantify the additional possible metrics, we have made the following observations. First, numerous software tools support the ESRI Shapefile format whereas not many tools work with Digital Line Graphs or TIGER files. Second, the amount of time we have spent implementing the LLS, DLG and TIGER file format loaders was increasing in the order of the listed file formats. We hypothesize that the increase is almost linear but it becomes quadratic as the file format is too complex to track and eliminate software bugs. Finally, the cost of storage and RAM has been rapidly decreasing over the last decade. We could not foresee the future technological advancements of storage media that would favor one file format over another.

7 Future Directions

One would like to incorporate the effects of computer clusters and mass storage systems on the storage versus boundary retrieval efficiency tradeoff evaluations for LLS, TIGER and DLG data structures. Our tradeoff study thus far has been in an isolated workstation environment. The results of our tradeoff study could differ when a very large cluster or mass storage system is used. We have identified several directions that further research could take.

First, investigate the effect of computer clusters on boundary retrieval efficiency assuming distributed or centralized locations of a large number of boundary files. The benefit of using a computer cluster would come from parallelization of loading and boundary reconstruction tasks.

Second, empirical results and theoretical analyses from our research thus far have shown file size (computer storage size) to be related to the amount of overlap between boundaries. The usage of a mass storage system will add to the time needed to load bytes from a file.

Another component of mass storage systems and

computer cluster environments is the Input/Output (I/O) bandwidth and I/O schemes. While the relationship between I/O bandwidth and boundary retrieval efficiency is straightforward (linear dependency), there are a few questions to ask about I/O schemes. For instance, can more efficient I/O schemes be used to improve boundary retrieval? Would message passing interface input/output (MPI-IO) have any effect? What would be the bottlenecks?

Finally, our ultimate goal is to understand multiple effects of electronic vector files on the archival process. We could mention just a few effects, such as vector file format, data organization and representation, algorithmic parallelization, scalability of vector file loading in terms file size and centralized or distributed file locations, software re-usability, computer platform dependency, computer cluster environments, I/O bandwidth and I/O schemes, and mass storage systems.

Acknowledgements

This research was supported by a National Archive and Records Administration (NARA) supplement to NSF PACI cooperative agreement CA #SCI-9619019. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, the National Archives and Records Administration, or the U.S. government.

References

- [1] Campbell, James B. Introduction to Remote Sensing, Second Edition. The Guilford Press, New York. 1996.
- [2] Miller, Catherine L. TIGER/Line Files Technical Documentation. UA 2000. U.S. Department of Commerce, Geography Division, U.S. Census Bureau.
<http://www.census.gov/geo/www/TIGER/TIGERua/ua2ktgr.pdf>
- [3] ArcExplorer, ESRI web site: <http://www.esri.com>
- [4] ESRI Shape file, File Format Specification, <http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf>
- [5] Bajcsy P. et. al., “Image To Knowledge”, documentation at web site:
<http://alg.ncsa.uiuc.edu/tools/docs/i2k/manual/index.html>.
- [6] Alumbaugh T.J. and Bajcsy P., “Georeferencing Maps with Contours in I2K”, ALG NCSA technical report, alg02-001, October 11 2002.
- [7] Groves P., S. Saha and P. Bajcsy, “Boundary Information Storage, Retrieval, Georeferencing and Visualization,” Technical Report NCSA-ALG-03-0001, February 2003.
- [8] “Data To Knowledge”, software documentation at

web site:

<http://alg.ncsa.uiuc.edu/tools/docs/d2k/manual/index.html>

- [9] "Digital Line Graph References" prepared by the Office of Information Technology,
<http://www.oit.ohio.gov/SDD/ESS/Gis/DigitalLineGraphs.aspx#Documentation>
- [10] Digital Line Graphs from 1:24,000-Scale Maps Data Users Guide I,
<http://www.geodata.gis.state.oh.us/dlg/usrguide/usrguide.htm>
- [11] "US GeoData, Digital Line Graphs" prepared by the U.S. Department of the Interior and the U.S. Geological Survey,
http://www.usgsquads.com/downloads/factsheets/usgs_dlg.pdf
- [12] Clutter D. and P.Bajcsy. "Storage and Retrieval Efficiency Evaluations of Boundary Data Representations for LLS, TIGER and DLG Data Structures," Technical Report NCSA-ALG-04-0007, October 2004.

Demonstration and Poster Abstracts



ABBYY Software House OCR, Forms Processing and Data Capture Technology Review

This abstract will provide an overview of ABBYY's applicable technologies for possible SDIUT05 product demos.

ABBYY Overview

ABBYY is a world leader in the development of recognition, data capture and linguistic technologies. Leading products include the award-winning FineReader line of OCR/ICR/OMR/Barcode recognition software and SDK, the FormReader line of form-processing solutions, and FlexiCapture Studio for data-capturing from various semi-structured documents and forms.

ABBYY FineReader OCR and Fraktur XIX

ABBYY's award-winning FineReader OCR technology converts static paper documents and PDF files into manageable electronic data.

ABBYY FineReader Engine is the most comprehensive data capture and document recognition / conversion SDK. It includes all the technologies needed for developing state-of-the-art intelligent data capture, document recognition, and document conversion systems. In addition to OCR, ICR, OMR and barcode recognition, it also offers image pre- and post-processing, document layout analysis, PDF conversion, and forms processing (for both fixed and semi-structured forms) technologies.

ABBYY FineReader Engine 7.1 offers all of the essential data capture and document conversion tools in a single toolkit. As a result, developers and integrators benefit from working with a single and efficient environment. Developers no longer need to outsource different technologies, learn different programming environments or deal with multiple licensing agreements. In addition, FineReader Engine's modular approach in its run-time licenses ensures that developers only pay for the technology they use in the end application.

ABBYY FineReader Engine is a common platform for ABBYY's recognition technology. FineReader Engine covers all the core technology functions that are inside of ABBYY's flagship products including ABBYY FineReader OCR, ABBYY FormReader and ABBYY FlexiCapture Studio. It also has additional recognition functions specifically designed for international conversion or capturing projects, such as Fast Mode Recognition, Document Analysis for Invoices, and CJK OCR (Recognition of Chinese, Japanese, and Korean).

FineReader Engine is flexible enough to support development of an application of any architecture and scale. Developers can create a client workstation designed from scratch, build a server-based solution, or integrate FineReader technologies into an existing application.

The First Omnifont OCR Fraktur: Ancient Text Recognition without Extensive Training

In January ABBY announced the availability of the first omnifont OCR software for recognition of “Fraktur” or Black Letter print, ABBYY FineReader XIX. More than two years in development, ABBYY FineReader XIX is designed to read ancient texts using “Fraktur” or Black Letter print and those printed between 1600 and 1938. It supports German, English, French, Italian and Spanish languages. FineReader XIX reads Fraktur, Schwabacher and many types of Texture (Gothic) fonts including Roman-type characters no longer in use such as the elongated “s” used in early English or French texts.

Until now, recognition systems have required many hours or days of training to recognize key words and symbols. ABBYY designed FineReader XIX to be an omnifont system for Fraktur and old European language recognition, allowing users to scan and read documents using Fraktur scripts and convert them into digital text with minimal training and dictionary work. ABBYY achieved this by developing a special classifier for Fraktur and special language dictionary models for Fraktur together with ABBYY’s partner ATAPY software. With the aim of delivering high levels of accuracy, ABBYY FineReader XIX has been trained using hundreds of sample documents, including 10 dictionaries and more than 105 books published between 1808 and 1930. In addition, 31,000 pages of text from different sources were used in a test base. Special recognition dictionaries using historical printed dictionaries representing the orthography of the 19th centuries for German were also developed.

ABBYY FormReader

FormReader, ABBYY’s powerful data capture and forms processing solution, is designed to extract information from paper forms and transfer captured data to databases or information management systems.

ABBYY FormReader is an easy-to-use form-processing application designed to extract information from paper forms and transfer captured data to databases or information management systems. FormReader is designed to recognize 177 machine-print and 16 handprint languages, checkmarks and barcodes. It supports XML, multiple file formats and databases as well as allows customizable data export through Automation. FormReader’s built-in form/template designers help reduce efforts in form definition dramatically and increase machine readability and data recognition. A user-friendly interface, including “Scan&Read”, guides users through steps of form processing -- scanning, template matching, recognition, verification and export. ABBYY FormReader offers organizations a fast-deployment and cost-effective solution without requiring extensive IT resources.

ABBYY FlexiCapture Studio Powerful Data Extraction for Semi-Structured Forms and Documents

ABBYY FlexiCapture Studio, backed by ABBYY’s latest FlexiCapture technology, is an add-on option for both the ABBYY FormReader line of form processing solutions and the

ABBYY FineReader line of OCR/ICR/OMR/barcode recognition to process semi-structured forms and documents. With FlexiCapture Studio and FormReader or FineReader Engine, users can automatically recognize information from various semi-structured forms and documents. It allows system developers to integrate this powerful and flexible tool to build or customize their own automated data-capture applications for processing semi-structured forms and documents.

Unlike other semi-structured forms processing technologies, FlexiCapture offers a GUI-based design environment versus a script-based one. This allows users to create a logical description of any semi-structured form or document, then use that description, or FlexiLayout, with an enabled capture application to simplify and expedite the processing task. With added support for new layout formats and pre-processing functions, the latest version of FlexiCapture further increases accuracy in analyzing and reading multiple form layouts, making it an ideal tool for processing multiple types of semi-structured forms and documents.

VERUS

A Middle Eastern Language OCR

Steven G. Schlosser
NovoDynamics, Inc.
123 N. Ashley St. STE 210
Ann Arbor, MI 48104
steve@novodynamics.com

Demonstration Abstract

VERUS is a recently released standalone OCR application for Middle Eastern languages. Four languages, Arabic, Farsi, Dari and Pashto, are recognized by the system as is embedded English and French. The software allows the user to scan individual pages or entire documents from within the application. Degraded pages are automatically cleaned in batch mode without a need for page-by-page parameter adjustment. Page skew and incorrect orientation (e.g., upside down) are automatically corrected. The software can automatically determine the language of a page to circumvent the need for manual pre-sorting. Pages selected for recognition are converted to Unicode text and can be saved in RTF format. Input formats include TIF, JPEG and PDF.

VERUS Professional includes an Application Programmer Interface (API) or SDK for users who wish to integrate the software into a third-party solution. The API allows access to intermediate results for use in controlling a workflow application. Examples of such uses might be integration of cleaned images into a workflow application's interface to improve readability during exploitation, or use of the language identification information to determine proper routing to a linguist. Many other options are also available to meet alternative application requirements. In addition to the output formats of VERUS, the Professional edition includes PDF with hidden text as an option. This enables users to create Middle Eastern language document repositories that can be searched with keywords and key phrases using Adobe Reader.

The VERUS demonstration covers the following typical sequence of software operations: loading an image or workspace, selecting language options, recognition, viewing and editing of output text, display and editing of text zones, export of text and saving of image and text files, and searching PDF with hidden text files.

NovoDynamics, Inc. is an Advanced Image Discovery company located in Ann Arbor, Michigan that specializes in processing, analyzing and retrieving critical information from both pristine and degraded sources. The company currently sells three products, ArborScript™ CRS, ArborScript™ ES and VERUS. ArborScript™ CRS is an SDK for Middle Eastern language OCR for Arabic, Farsi, Dari, and Pashto. ArborScript™ ES is a document exploitation system for Middle Eastern languages that supports image capture, OCR, full-text indexing, and search and retrieval. VERUS is an advanced Middle Eastern language OCR with an intuitive graphical user interface. NovoDynamics' products are differentiated by their superior accuracy and their ability to successfully process large volumes of "real-world" documents, such as smudged faxes, poor quality photocopies, and documents with stains. As a result of NovoDynamics' accomplishments and technology, In-Q-Tel, a venture capital firm funded by the CIA, has made a significant investment in the company and has selected NovoDynamics as an In-Q-Tel portfolio company.

IDG: A Business Information Extraction, Management, and Routing Front-End for Content Management Systems

Vikas Krishna, Savitha Srinivasan, Neil Boyette, Isaac Cheng,
Jeffrey Kreulen and Tapas Kanungo
IBM Almaden Research Center
650 Harry Road, San Jose, CA 95120
{vikas,savitha,isaacc,nboyette,kreulen,kanungo}@us.ibm.com

ABSTRACT

Content management systems are becoming the standard tool for storing, organizing, and managing corporate information. One of the driving forces behind this is compliance with government regulation. The other driver is pure economic – it is less expensive to manage and manipulate information electronically via content management systems than have it spread over various forms of information systems.

While content management systems are efficient in managing information once it is in the system, getting the information into the content management system is a manual, time-consuming, error-prone, and expensive process. In this article we describe the Intelligent Document Gateway (IDG), which is an information extraction, management and routing front-end for content management systems. IDG can be tailored to extract information from various media and formats, including document images.

Keywords

Information extraction, routing, OCR, business transformation, content management, workflow.

1. INTRODUCTION

Government regulations like HIPAA, Sarbanes-Oxley, Patriot Act, etc., are requiring companies to migrate their corporate information into electronic content management systems. A typical content management (CM) system stores, organizes, manages, and searches information in various formats like relational databases, web pages, voice and email messages, FAX documents, power point and PDF documents, instant messages, etc. Besides the basic functionalities, CM systems are also responsible for providing a) records management, which is responsible for destroying documents as soon as the government-specified retention period is over, or until a time specified by the business retention policy, b) encryption and access control, c) workflow specification, which organizes tasks into a sequential order, and d) storage and archival.

Today CM systems have become quite crucial for businesses to function. However, the CM market is quite fragmented with products like IBM Content Manager, FileNet, Documentum/EMC, InterWoven, Stellent, Mobius, OpenText, etc., each with its proprietary data representation and information exchange formats. Open standards like JSR 170, and their subsequent adoption by the software vendors will eventually make CM systems modular, interoperable, and standardized.

The Achilles heel of CM systems, however, is data entry. While CM systems can handle information that is already in the system, populating CM systems with various types of data is currently out of the scope of CM systems, and is done manually. Manual data entry can be laborious, time consuming, error-prone and expensive. For example, any logic associated with the inbound and outbound document flow as well as the indexing is commonly hard-coded in specific applications. This makes it difficult to identify and change the document-routing logic as required by dynamic business scenarios.

In this article we describe the architecture of the Intelligent Document Gateway (IDG) that works as a front end to the IBM Content Manager and allows users to ingest new information into IBM CM by applying user-specified business rules to the information and then routing the information to the appropriate repository in IBM CM. Metadata is extracted from the incoming documents (e.g. form images) or messages using rules that are authored by the user. The rules are expressed in spreadsheets enabling business users to author and update rules using a familiar paradigm. Updated rules are deployed into a rule engine that handles rule validation and execution. Rule generation is automated using automatic generation of XPath expressions and business rules are validated using XML schema validation. This system is currently deployed in various IBM divisions and potential applications being considered are: citizenship and immigration services, import compliance, commercial invoice management, customs and border protection, etc.

2. RELATED WORK

The notion of routing FAXes has been proposed in the OCR research literature by many researchers (e.g. [L01]). However, these systems focus only on the identification of names and addresses of people, and do not address the extraction of various other business variables like contract amount, point of contact, etc. and CM system issues like retention, encryption, etc. In the commercial world, Kofax distributes Ascent Capture software that extracts metadata from document images. The output can also be integrated into various CM systems including the IBM CM system. However, Ascent Capture extracts text only; application of business logic to the extracted has to be achieved outside the system. Furthermore, the system is setup to work in a batch mode and does not have the dynamic/realtime updating and authoring facilities.

Document management with workflow has been prevalent for several decades and many document management systems incorporate this feature. Aalst et al [AVK01] developed a

workflow language named XRL (eXchangeable Routing Language) for supporting cross-organizational processes. XRL also uses XML for the representation of process definitions and Petri nets for its semantics. Since XRL is instance-based, workflow definitions can be changed on the fly and sent across organizational boundaries. However, these features make cross-organizational workflows susceptible to errors. They also present XRL/Woflan, a software tool using state-of-the-art Petri-net analysis techniques for verifying XRL workflows. The tool uses XSLT (Extensible Style Language Transformations) to translate XRL specifications to a specific class of Petri nets to determine whether the workflow is correct.

In the area of Business Rule Processing (BRP), OMG issued a RFP for Business Semantics of Business Rules [OMG]. The proposal aims to enable business people to define rules in their business languages, describe artifacts for their business domains, and make rule definitions clear, unambiguous and portable to other representations. Domain knowledge is usually encoded in rules and an inference engine makes decisions based on these rules [DHW94]. OPS5 [BFKM85] is one of the earliest expert system rule languages developed at Carnegie-Mellon University. It uses a forward chaining inference engine and programs search working memory to find matching rules to execute. Other well-known expert systems include Java Expert System Shell (JESS) [JESS] from Sandia National Labs, and Agent Building and Learning Environment (ABLE) [BSPMD02] from IBM. Our work is based on ABLE as our rule processing engine.

3. IDG SYSTEM ARCHITECTURE

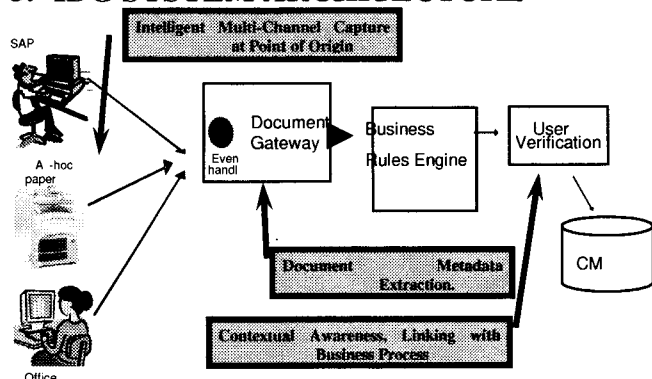


Figure 1. System Architecture

In Figure 1 we have outlined the basic architecture of our system. The intelligent document gateway (IDG) receives messages from the Enterprise Service Bus (ESB) via Java Messaging Services (JMS) and extracts business information from the messages. These information chunks are then either used to either trigger business rules, or are themselves information chunks that get moved to certain locations in the content management system. The user has ability to verify the business logics that are triggered, and the system also allows the user to author new business rules or modify existing ones. During the execution of the rules, the actual input parameters come from each purchase order, which is an XML document. Rule actions are embedded in XML, compressed, and sent to an appropriate outbound ESB; they can vary broadly depending on the customer and location of a purchase order.

3.1 Information Routing Architecture

As shown in Figure 2, the architecture provides the Intelligent Document Gateway with the flexibility to work with any document routing logic and any inbound document type (e.g. ASCII, XML, or form images). Both the business logic and the structure of inbound documents are externalized by the gateway in standard formats. The business logic is externalized as a comma-separated value (CSV) file. This allows business users to easily author and edit it with a familiar spreadsheet tool.

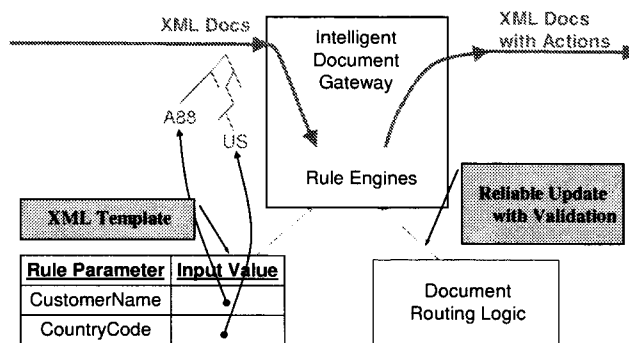


Figure 2. Document Routing Architecture

Each rule can have a number of input parameters, whose values are to be determined at runtime. The link between each parameter name and the location of its runtime value in each inbound document is externalized as a XML template file per document type. This allows the gateway to work with any inbound document type which in turn can have any internal document structure.

3.2 Document Routing Authoring

Business users can author and edit the routing rules with a familiar spreadsheet tool, as shown in Figure 3. In particular, business users do not need to write any computer programs, macros, or scripts. The knowledge base format contains just the essential elements of the business domain. The rule format supports logical operators (AND, OR, NOT), pattern matching, and set operations (membership and complement). For instance, a business user can specify that a country code should not be in the set {UK, SE}.

RuleId	CustomerId	Logic	CustomerName	CountryCode	Action	CSRUsers	CSRManagers	Bcc
1	3MDenmar	AND1000142262			Email:	ruif@uk.ibm.com	ED@uk.ibm.com	custsupi@ibm.com
2	ABBUSCSA	AND100001	AND "ABB"		Email:	smoeller@ibm.com	b2beron@ibm.com	custsupi@ibm.com
3	ABBUSCSA	AND100001	AND "ABB"	UK SE	Email:	ABBORDERjfs@fr.ibm.com	comjru	custsupi@ibm.com
4	ADPCSRRA	AND100004405			Email:	donissmit@ibm.com	b2beron@ibm.com	custsupi@ibm.com
5	AetnaUSC	AND100001	AND "Aetna"	US	Email:	dickpunk@ibm.com	comjlebuffe	custsupi@ibm.com
6	AlcoaCSR	AND100001497			Email:	alcoa@us.ibm.com	b2beron@ibm.com	custsupi@ibm.com
7	AMEXCSR	AND10000C	AND "American Express"		Email:	dkihman@ibm.com	kgkluv@ibm.com	custsupi@ibm.com

Figure 3. Document Routing Logic Authoring

3.3 Reliable Update with Validation

Any routing rule can be updated while the Intelligent Document Gateway is in service. The gateway receives updates in a staging area, where it checks for syntactic and semantic errors. If the new logic has a problem, the gateway tries to identify the rules that might have caused the problem. It then notifies the business users of the problem by email, along with any metadata from the inbound document to help them understand the problem. The updates are automatically delivered to the Intelligent Document Gateway from a secure repository via an encrypted channel with

public-key authentication to protect customers' privacy without sending the password over the network.

Behind the scenes, the knowledge base is processed by the scripting engine of ABLE. This allows our architecture to also externalize the powerful rule-set interface of ABLE for advanced document engineers. Each rule set can contain any number of rule blocks where each rule contains a number of input parameters. Each parameter is bound to a value from an inbound XML document at runtime based on an XML template which defines the linkage between each parameter and the location path to an attribute or an element in an inbound XML document.

3.4 XML Invoice Image Template Authoring

We provide an Eclipse plug-in called the Document Gateway Architect, for designers to author and edit XML templates for extracting fields from images of invoice documents as shown in Figure 4. The template is used to extract fields, which are processed by ABBYY OCR to extract symbolic text.

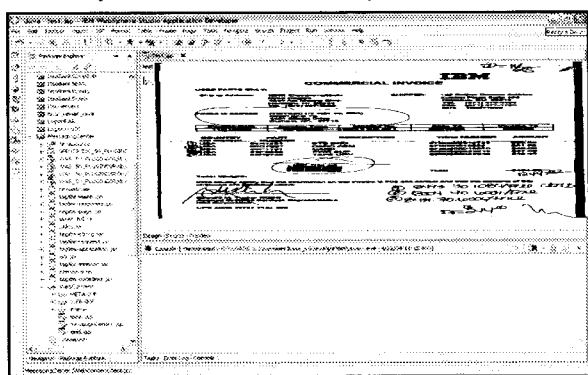


Figure 4. XML Invoice Image Template Authoring and OCR

Business processes are organized in Eclipse projects along with the business rules in them. To edit business rules, a user simply double-clicks on a rule file to launch Excel within Eclipse. The Document Gateway Architect validates business processes with XML Schema definitions. The gateway uses the standard XSLT engine to process the XML template to provide business rules with the runtime values of input parameters.

Authoring or editing an XML template does not require any knowledge of XML or XPath. With the Document Gateway Architect, a designer imports an inbound document into a business process. The tool then displays a familiar expandable tree view for the designer to browse the XML document, as illustrated in the top half of Figure . When the designer selects the attribute or element that contains the input value of a business rule, the tool automatically generates the correct XPath expression.

4. IMAGING CHALLENGES

Currently we are working on many challenges that arise during information extraction from scanned document images. Some of these include:

- Information extraction from geometric forms where the fields of the form are always at the same physical location in the original form template but are skewed during the scanning process [KH99, KK02].
- Information extraction from boiler plate forms that have running prose, but certain entities in the document

are replaced with new strings (e.g. names and project duration in contractual documents) [KM03].

- Extraction of handwritten entries in geometric forms.
- Extraction of information of faxed geometric and boiler-plate forms that have very low quality due to multiple re-faxings.
- Multilingual extraction for international transactions (e.g. immigration, import and export, customs, etc.) that require multilingual text recognition.
- Benchmarking dataset for OCR-based information extraction.

5. CONCLUSION

We described the architecture of Intelligent Document Gateway (IDG), which is an information extraction, management and routing front-end for document management systems. We also described some of the challenges we are currently facing in extracting information from various document image sources. This system is currently deployed in various IBM divisions and we are currently working on extending it to various applications in customs, citizenship and immigration, border protection, etc.

6. REFERENCES

1. [AVK01] W.M.P. van der Aalst, H.M.W. Verbeek, and A. Kumar. *XRL/Woflan: Verification and Extensibility of an XML/Petri-net based language for inter-organizational workflows*. In Proceedings of the 6th INFORMS Conference on Information Systems and Technology, 2001.
2. [BSPMD02] J. P. Bigus, D. A. Schlosnagle, J. R. Pilgrim, W. N. Mills, and Y. Diao. "ABLE: A Toolkit for Building Multiagent Autonomic Systems," *IBM Systems Journal*, Vol. 41.No.3, pp. 35071, 2002.
3. [BFKM85] L. Brownston, R. Farrell, E. Kant, and N. Martin. *Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming*. Reading, Massachusetts: Addison-Wesley, 1985.
4. [DHW94] U. Dayal, E. N. Hanson, and J. Widom. "Active Database Systems," in *Modern Database Systems: The Object Model, Interoperability, and Beyond*, W. Kim, Ed.: Addison-Wesley, September 1994, pp. 434-456.
5. [JESS] E. Friedman-Hill. *Jess in Action*: Manning Publications, July 2003.
6. [KS04] V. Krishna and S. Srinivasan, Towards Smarter Documents. In Proceedings of CIKM 2004.
7. [OMG] "Business Semantics of Business Rules (BSBR) RFP," OMG Business Enterprise Integration Domain Task Force June 2003.
8. [L01] L. Likforman-Sulem, "Name block location in facsimile images using spatial/visual cues," *Proc. ICDAR*, pp. 680-684, Sept. 2001.
9. [KH99] T. Kanungo and R. M. Haralick, "A closed-loop methodology for generating character groundtruth for scanned images," *IEEE PAMI*, vol. 21, pp. 179-183, 1999.
10. [KK02] D. Kim and T. Kanungo, "Attributed point matching for automatic groundtruth registration," *IJDAR*, vol. 5, pp.47-66, 2002.
11. [KM03] T. Kanungo and S. Mao, "Stochastic language models for style-directed physical layout analysis of documents," *IEEE PAMI*, vol. 12, pp. 583-596, 2003.

Fast skew and *slant* correction for Arabic written word or line

Essam M. Zaki
Sakhr Software USA, Inc.

Mohamed El-Adawi
Helwan University, Faculty of Engineering

Abstract

Word or line processing is important stage in OCR (Optical Character Recognition.) or ICR (Intelligent Character Recognition.) systems. Here, we propose two algorithms appropriate for this stage. The first one corrects the skewing of word (line). The second removes the slant from printed or handwritten word (line). The skewing detection algorithm collects the first foreground pixel in every image column then use linear regression to find the best fitting line, the slope of the line is the skew angle. The slant detection algorithm uses the connected component analysis, the slant angle is calculated by finding the edged image and removing horizontal lines then finding the slant histogram for connected components, the maximum peak in slant histogram will be the slant angle. The algorithms have been tested for printed and handwritten words (lines) of Arabic and English languages. The algorithms provide high accuracy and fast results. The overall accuracy of OCR or ICR systems have been improved after using skew and slant correction.

Evaluation Workshop Abstracts

Performance Evaluation of Multilingual Document Exploitation Systems

Steven G. Schlosser
NovoDynamics, Inc
123 N. Ashley St. STE 210
Ann Arbor, MI 48104
steve@novodynamics.com

Abstract

This presentation identifies and discusses key forms of performance evaluation called for by multilingual document exploitation systems. The architecture and capabilities of ArborScript™ ES are used to highlight typical information requirements of exploitation system test data. RML, an XML-based recognition markup language used by ArborScript ES, is offered as an example of how comprehensive groundtruth can be structured. In addition, common practical issues in formulating groundtruth are described.

Issues with Automatic OCR Evaluation and Metrics

Kristian J. Concepcion
7525 Colshire Drive – M/S H305
McLean, VA 22102

Abstract

In this presentation, we discuss the issues that arose while semi-automatically creating a degraded Arabic corpus for OCR testing, and the subsequent weakness of current OCR metrics to report on the complexity of the documents to be processed. Using a hybrid of manual and automatic techniques, we were able to create clean and degraded versions of our test corpus, with many controlled variables (dpi, font, point size, font face). Character accuracy and word accuracy were then calculated against the ground truth. However, while character accuracy and word accuracy are useful measures of the performance of an OCR engine, they are neither descriptive of how the OCR engine handles document features such as images, watermarks, or logos, nor do they differentiate between the cause of extraneous characters that appear in the output. While these shortcomings are partly due to a lack of ground truth standards for how to ground truth a complex document, they also occur because these accuracies cannot encapsulate the numerous issues that arise from processing real-world documents.

Evaluation Issues in ImageRefiner

Kristen Summers Eugene Borovikov

CACI

{ksummers,eborovikov}@caci.com

Abstract

ImageRefiner is an application that uses machine learning to select enhancement methods to apply to document images, for the purpose of improving OCR accuracy. In addition to requiring OCR evaluation for both the training phase and testing the system, it raises additional evaluation issues. These include: appropriate evaluation of selected enhancement methods for particular document images; evaluation of segmentation of images into regions for separate enhancement; and evaluation of the generality of the system, including both the effects of a training set that imperfectly represents the application data, as is often unavoidable in operational environments, and the effects of heterogeneity within that training set.

The Sporadic Nature of OCR Evaluations

Tapas Kanungo

IBM Almaden Research Center
650 Harry Road, San Jose, CA 95120
kanungo@us.ibm.com

ABSTRACT

The Optical Character Recognition (OCR) community has had a long history: the OCR problem is one of the oldest pattern recognition problems that researchers have worked on, and commercial OCR products have been around for decades. However, a long history does not imply maturity. The maturity of a field is typically reflected in, amongst other things, the existence of a regular objective evaluation of the state of the art. For example, i) the processor design community has been running the SPEC benchmarks for decades; ii) the database community has been evaluating database performance using various TPC benchmarks; iii) the information retrieval community has been running the TREC evaluations for a very long time, and iv) the speech community has its regular evaluations. However, in the case of OCR, we can not claim that we have had continued evaluations over decades. The UNLV evaluations of OCR systems were held for five years. Arabic and multilingual system evaluations were held at University of Maryland for two years. There was an OCR track at TREC one year, and more recently, the VACE program had video OCR evaluations. On the positive side, the community has produced various resources over the years: i) the UNLV evaluation datasets and tools, ii) the UW dataset and tools, iii) the UMD groundtruthing tool PSET, the logo dataset, and the Viper tool. At this evaluation workshop, we should i) identify how we can establish a long running evaluation program, ii) agree on what kinds of OCR problems and evaluations are necessary for the government, academia and industry, and iii) identify what organization is best positioned to run the evaluations on a yearly basis.

Ground Truth Representation used in Testing and Optimization of the Optical Word Recognition System

**Mike Ladwig
Northrop Grumman Corporation**

Abstract

This paper describes the ground truth representation and associated software tools used in testing and optimizing the Optical Word Recognition system. Optical Word Recognition (OWR) technology addresses key elements in the process of analyzing scanned document page images using an optical correlation approach. OWR applications have focused on document triage - the rapid identification of high value documents based on content such as signatures, graphical logos, and keywords. In these applications, OWR is used to identify high-value items rather than perform general content conversion. Because of this difference, OWR testing and optimization uses a ground truth representation oriented towards the representation of metadata about the page as whole and interesting objects found on it. This representation is used throughout the suite of software used to test and optimize OWR.

Metrics for Word Spotting

Sargur Srihari

CEDAR, University of Buffalo

ABSTRACT

Central to searching a document repository is a method for finding a query word in each of the documents. In the case of scanned handwritten documents there is uncertainty in being able to find words since their shapes can vary greatly. Thus there is a level of uncertainty in word spotting-- unlike in the case of electronic text where words can be found with high accuracy by using string matching algorithms. In this paper we propose measures for the evaluation of a word spotting system for handwritten documents. It consists of measures for line segmentation, word segmentation and word matching. The use of these measures in evaluating the performance of the CEDARABIC system in spotting Arabic words is illustrated.

Evaluating with Informational Confidence

Stefan Jaeger
David Doermann

Institute for Advanced Computer Studies
University of Maryland,
College Park, MD 20742, USA
jaeger@umiacs.umd.edu

Abstract

A fair comparison is the key to a meaningful evaluation of different approaches to the same problem. In order to ensure a fair evaluation, however, it is not sufficient to concentrate solely on the recognition rates alone. We need to take into account the confidence values, which indicate how confident a classifier is in its output results. Confidence values usually play an important role in post-processing and are essential to finding rejection thresholds or ranking documents according to a given objective function. For instance, a classifier that assigns the correct label to a test pattern with high confidence should be identified as being superior to another classifier giving the same correct label but with lower confidence.

We are promoting an information-theoretical technique to meet these requirements. In particular, we assume that each confidence value conveys information that is directly related to the performance of its classifier in the application domain. In an evaluation step, we can normalize each confidence value so that it equals its informational content. These newly computed confidence values, called *informational* confidence values, will then replace the old, raw confidence values. The normalization technique makes comparison of confidence values straightforward and even allows combination of multiple approaches into a single integrated system. Informational confidence values have already shown their effectiveness in several pattern recognition problems, such as character recognition or script identification.

Author Index

-A-

Ablavsky, Vitaly.....49
Agam, G.23
Al-Onaizan, Yaser.....103
Argamon, S......23

-B-

Babu, Pavithra123
Bajcsy, Peter211
Bhole, Chetan.....123
Boyette, Neil.....231
Borovikov, Eugene79, 243
Borsack, Julie.....39

-C-

Chang, Richard.....187
Chen, K.31
Cheng, Isaac231
Clutter, David211
Concepcion, Kristian J......241

-D-

Doermann, David.....31, 251

-E, F-

El-Adawi, Mohamed235
Farooq, Faisal89, 103
Femiani, John C......65
Fisher, Francis.....187
Frieder, O.23

-G-

Gantz, Donald T......173, 197
Govindaraju, Venu.....89
Grossman, D.23

-H-

Haritaoglu, Esin Darici55
Haritaoglu, Ismail.....55
Henderson, Thomas C......157
Holden, Steve.49

-I, J-

Jaeger, Stefan.....31, 251

-K-

Kanungo, Tapas231, 245
Kreulen, Jeffrey.....231
Krishna, Vikas.....231

-L-

Ladwig, Mike.....247
Lewis, D.23
Lin, Yun.....11
Lopresti, Daniel111

-M-

MacRostie, Ehry.....15
Marcus, Kevin.....187
Miller, John J......197
Monnier, Camille49

-N-

Nagy, George.....111
Nartker, Tom.....39
Natarajan, Premkumar15

-O, P-

Phielipp, Mariano65
Price, Robert.....97

-Q, R-

Razdan, Anshuman.....65

-S-

Sarkar, Prateek167
Saund, Eric.....167

<i>Schlosser, Steven G.</i>	139, 229, 239
<i>Seales, W. Brent</i>	11
<i>Shalev, Michael</i>	71
<i>Snorrason, Magnus</i>	49
<i>Spitz, A. Lawrence</i>	133
<i>Sridharan, Karthik</i>	89
<i>Srihari, Sargur</i>	123, 249
<i>Srinivasan, Harish</i>	123
<i>Srinivasan, Savitha</i>	231
<i>Summers, Kristen</i>	243

-T-

<i>Taghva, Kazem</i>	39
<i>Turner, Joi</i>	187
<i>Turner, Mark</i>	79

-U, V-

<i>Valenzuela, Imelda</i>	225
<i>Vincent, Luc</i>	9
<i>Vogt, Robert C.</i>	139

-W-

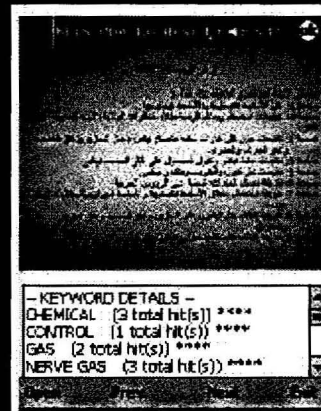
<i>Walch, Mark A.</i>	173, 197
-----------------------------	----------

-X, Y, Z-

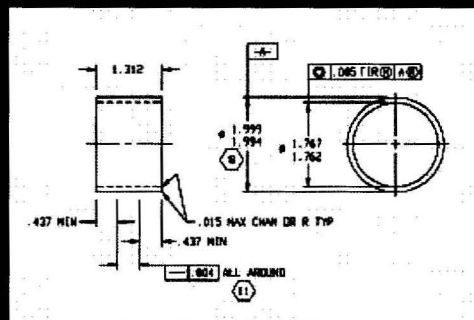
<i>Xu, Chimiao</i>	157
<i>Yaghi, Jim</i>	133
<i>Zaki, Essam M.</i>	235
<i>Zavorin, Ilya</i>	79
<i>Zhu, Guangyu</i>	31
<i>Zukas, Anthony</i>	97

String candidate	Linguistic score	Gloss	Details
Alsbak almshtfyn (الشمك المسترين)	12	The young experts	valid words, correct phrase structure - well formed agreement
Alsbak almshtfyn (الشمك المسترين)	23	ungrammatical (*experts the windows)	valid words, incorrect grammatical context
Alsbkk almshtfyn (الشمك المسترين)	-	word discarded	invalid word.

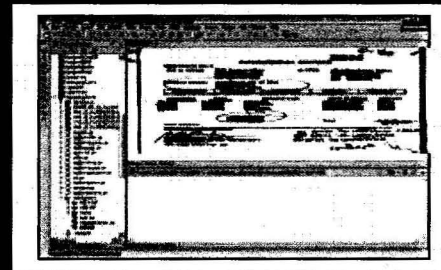
Processing of Arabic Documents



Document Analysis Applications



Cross Domain Applications



Demonstrations



SDIUT'05

2005 Symposium on Document
Image Understanding Technology

Marriott
Inn and Conference Center
College Park, Maryland
November 2-4, 2005

