# Online Large-Margin Training of Syntactic and Structural Translation Features

**David Chiang**
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292, USA
chiang@isi.edu

**Yuval Marton** and **Philip Resnik**
Department of Linguistics and the UMIACS
Laboratory for Computational Linguistics
and Information Processing
University of Maryland
College Park, MD 20742, USA
{ymarton,resnik}@umiacs.umd.edu

## Abstract

Minimum-error-rate training (MERT) is a bottleneck for current development in statistical machine translation because it is limited in the number of weights it can reliably optimize. Building on the work of Watanabe et al., we explore the use of the MIRA algorithm of Crammer et al. as an alternative to MERT. We first show that by parallel processing and exploiting more of the parse forest, we can obtain results using MIRA that match or surpass MERT in terms of both translation quality and computational cost. We then test the method on two classes of features that address deficiencies in the Hiero hierarchical phrase-based model: first, we simultaneously train a large number of Marton and Resnik's soft syntactic constraints, and, second, we introduce a novel structural distortion model. In both cases we obtain significant improvements in translation performance. Optimizing them in combination, for a total of 56 feature weights, we improve performance by 2.6 Bleu on a subset of the NIST 2006 Arabic-English evaluation data.

## 1 Introduction

Since its introduction by Och (2003), minimum error rate training (MERT) has been widely adopted for training statistical machine translation (MT) systems. However, MERT is limited in the number of feature weights that it can optimize reliably, with folk estimates of the limit ranging from 15 to 30 features.

One recent example of this limitation is a series of experiments by Marton and Resnik (2008), in which they added syntactic features to Hiero (Chiang, 2005; Chiang, 2007), which ordinarily uses no linguistically motivated syntactic information. Each of their new features rewards or punishes a derivation depending on how similar or dissimilar it is to a syntactic parse of the input sentence. They found that in order to obtain the greatest improvement, these features had to be specialized for particular syntactic categories and weighted independently. Not being able to optimize them all at once using MERT, they resorted to running MERT many times in order to test different combinations of features. But it would have been preferable to use a training method that can optimize the features all at once.

There has been much work on improving MERT's performance (Duh and Kirchoff, 2008; Smith and Eisner, 2006; Cer et al., 2008), or on replacing MERT wholesale (Turian et al., 2007; Blunsom et al., 2008). This paper continues a line of research on online discriminative training (Tillmann and Zhang, 2006; Liang et al., 2006; Arun and Koehn, 2007), extending that of Watanabe et al. (2007), who use the Margin Infused Relaxed Algorithm (MIRA) due to Crammer et al. (2003; 2006). Our guiding principle is practicality: like Watanabe et al., we train on a small tuning set comparable in size to that used by MERT, but by parallel processing and exploiting more of the parse forest, we obtain results using MIRA that match or surpass MERT in terms of both translation quality and computational cost on a large-scale translation task.

Taking this further, we test MIRA on two classes of features that make use of syntactic information and hierarchical structure. First, we generalize Marton and Resnik's (2008) soft syntactic constraints by training all of them simultaneously; and, second, we introduce a novel structural distortion model. We obtain significant improvements in both cases, and further large improvements when the two feature sets are combined.

The paper proceeds as follows. We describe our training algorithm in section 2; our generalization of Marton and Resnik's soft syntactic constraints in section 3; our novel structural distortion features in section 4; and experimental results in section 5.

## 2 Learning algorithm

The translation model is a standard linear model (Och and Ney, 2002), which we train using MIRA (Crammer and Singer, 2003; Crammer et al., 2006), following Watanabe et al. (2007). We describe the basic algorithm first and then progressively refine it.

### 2.1 Basic algorithm

Let $\mathbf{e}$, by abuse of notation, stand for both output strings and their derivations. We represent the feature vector for derivation $\mathbf{e}$ as $\mathbf{h}(\mathbf{e})$. Initialize the feature weights $\mathbf{w}$. Then, repeatedly:

- Select a batch of input sentences $\mathbf{f}_1, \ldots, \mathbf{f}_m$.
- Decode each $\mathbf{f}_i$ to obtain a set of hypothesis translations $\mathbf{e}_{i1}, \ldots, \mathbf{e}_{in}$.
- For each $i$, select one of the $\mathbf{e}_{ij}$ to be the *oracle translation* $\mathbf{e}_i^*$, by a criterion described below. Let $\Delta\mathbf{h}_{ij} = \mathbf{h}(\mathbf{e}_i^*) - \mathbf{h}(\mathbf{e}_{ij})$.
- For each $\mathbf{e}_{ij}$, compute the *loss* $\ell_{ij}$, which is some measure of how bad it would be to guess $\mathbf{e}_{ij}$ instead of $\mathbf{e}_i^*$.
- Update $\mathbf{w}$ to the value of $\mathbf{w}'$ that minimizes:

$$\frac{1}{2}\|\mathbf{w}' - \mathbf{w}\|^2 + C \sum_{i=1}^{m} \max_{1 \le j \le n} (\ell_{ij} - \Delta\mathbf{h}_{ij} \cdot \mathbf{w}') \quad (1)$$

  where we set $C = 0.01$. The first term means that we want $\mathbf{w}'$ to be close to $\mathbf{w}$, and second term (the *generalized hinge loss*) means that we want $\mathbf{w}'$ to score $\mathbf{e}_i^*$ higher than each $\mathbf{e}_{ij}$ by a margin at least as wide as the loss $\ell_{ij}$.

When training is finished, the weight vectors from all iterations are averaged together. (If multiple passes through the training data are made, we only average the weight vectors from the last pass.) The technique of averaging was introduced in the context of perceptrons as an approximation to taking a vote among all the models traversed during training, and has been shown to work well in practice (Freund and Schapire, 1999; Collins, 2002). We follow McDonald et al. (2005) in applying this technique to MIRA.

Note that the objective (1) is not the same as that used by Watanabe et al.; rather, it is the same as that used by Crammer and Singer (2003) and related to that of Taskar et al. (2005). We solve this optimization problem using a variant of sequential minimal optimization (Platt, 1998): for each $i$, initialize $\alpha_{ij} = C$ for a single value of $j$ such that $\mathbf{e}_{ij} = \mathbf{e}_i^*$, and initialize $\alpha_{ij} = 0$ for all other values of $j$. Then, repeatedly choose a sentence $i$ and a pair of hypotheses $j, j'$, and let

$$\mathbf{w}' \leftarrow \mathbf{w}' + \delta(\Delta\mathbf{h}_{ij} - \Delta\mathbf{h}_{ij'}) \quad (2)$$

$$\alpha_{ij} \leftarrow \alpha_{ij} + \delta \quad (3)$$

$$\alpha_{ij'} \leftarrow \alpha_{ij'} - \delta \quad (4)$$

where

$$\delta = \underset{[-\alpha_{ij}, \alpha_{ij'}]}{\text{clip}} \frac{(\ell_{ij} - \ell_{ij'}) - (\Delta\mathbf{h}_{ij} - \Delta\mathbf{h}_{ij'}) \cdot \mathbf{w}'}{\|\Delta\mathbf{h}_{ij} - \Delta\mathbf{h}_{ij'}\|^2} \quad (5)$$

where the function $\text{clip}_{[x,y]}(z)$ gives the closest number to $z$ in the interval $[x, y]$.

### 2.2 Loss function

Assuming Bleu as the evaluation criterion, the loss $\ell_{ij}$ of $\mathbf{e}_{ij}$ relative to $\mathbf{e}_i^*$ should be related somehow to the difference between their Bleu scores. However, Bleu was not designed to be used on individual sentences; in general, the highest-Bleu translation of a sentence depends on what the other sentences in the test set are. Sentence-level approximations to Bleu exist (Lin and Och, 2004; Liang et al., 2006), but we found it most effective to perform Bleu computations in the context of a set $O$ of previously-translated sentences, following Watanabe et al. (2007). However, we don't try to accumulate translations for the entire dataset, but simply maintain an exponentially-weighted moving average of previous translations.

More precisely: For an input sentence **f**, let **e** be some hypothesis translation and let $\{\mathbf{r}_k\}$ be the set of reference translations for **f**. Let $\mathbf{c}(\mathbf{e}; \{\mathbf{r}_k\})$, or simply $\mathbf{c}(\mathbf{e})$ for short, be the vector of the following counts: $|\mathbf{e}|$, the effective reference length $\min_k |\mathbf{r}_k|$, and, for $1 \leq n \leq 4$, the number of $n$-grams in **e**, and the number of $n$-gram matches between **e** and $\{\mathbf{r}_k\}$. These counts are sufficient to calculate a Bleu score, which we write as $\textsc{Bleu}(\mathbf{c}(\mathbf{e}))$. The pseudo-document $O$ is an exponentially-weighted moving average of these vectors. That is, for each training sentence, let $\hat{\mathbf{e}}$ be the 1-best translation; after processing the sentence, we update $O$, and its input length $O_f$:

$$O \leftarrow 0.9(O + \mathbf{c}(\hat{\mathbf{e}})) \tag{6}$$

$$O_f \leftarrow 0.9(O_f + |\mathbf{f}|) \tag{7}$$

We can then calculate the Bleu score of hypotheses **e** in the context of $O$. But the larger $O$ is, the smaller the impact the current sentence will have on the Bleu score. To correct for this, and to bring the loss function roughly into the same range as typical margins, we scale the Bleu score by the size of the input:

$$B(\mathbf{e}; \mathbf{f}, \{\mathbf{r}_k\}) = (O_f + |\mathbf{f}|) \times \textsc{Bleu}(O + \mathbf{c}(\mathbf{e}; \{\mathbf{r}_k\})) \tag{8}$$

which we also simply write as $B(\mathbf{e})$. Finally, the loss function is defined to be:

$$\ell_{ij} = B(\mathbf{e}_i^*) - B(\mathbf{e}_{ij}) \tag{9}$$

## 2.3 Oracle translations

We now describe the selection of $\mathbf{e}^*$. We know of three approaches in previous work. The first is to force the decoder to output the reference sentence exactly, and select the derivation with the highest model score, which Liang et al. (2006) call *bold updating*. The second uses the decoder to search for the highest-Bleu translation (Tillmann and Zhang, 2006), which Arun and Koehn (2007) call *max*-Bleu *updating*. Liang et al. and Arun and Koehn experiment with these methods and both opt for a third method, which Liang et al. call *local updating*: generate an $n$-best list of translations and select the highest-Bleu translation from it. The intuition is that due to noise in the training data or reference translations, a high-Bleu translation may actually use peculiar rules which it would be undesirable to encourage the model to use. Hence, in local updating, the search for the highest-Bleu translation is limited to the $n$ translations with the highest model score, where $n$ must be determined experimentally.

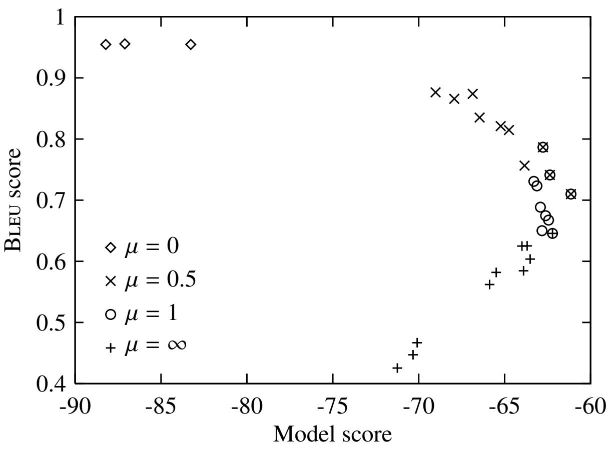

Figure 1: Scatter plot of 10-best unique translations of a single sentence obtained by forest rescoring using various values of $\mu$ in equation (11).

Here, we introduce a new oracle-translation selection method, formulating the intuition behind local updating as an optimization problem:

$$\mathbf{e}^* = \arg\max_{\mathbf{e}} (B(\mathbf{e}) + \mathbf{h}(\mathbf{e}) \cdot \mathbf{w}) \tag{10}$$

Instead of choosing the highest-Bleu translation from an $n$-best list, we choose the translation that maximizes a combination of (approximate) Bleu and the model.

We can also interpret (10) in the following way: we want $\mathbf{e}^*$ to be the max-Bleu translation, but we also want to minimize (1). So we balance these two criteria against each other:

$$\mathbf{e}^* = \arg\max_{\mathbf{e}} (B(\mathbf{e}) - \mu(B(\mathbf{e}) - \mathbf{h}(\mathbf{e}) \cdot \mathbf{w})) \tag{11}$$

where $(B(\mathbf{e}) - \mathbf{h}(\mathbf{e}) \cdot \mathbf{w})$ is that part of (1) that depends on $\mathbf{e}^*$, and $\mu$ is a parameter that controls how much we are willing to allow some translations to have higher Bleu than $\mathbf{e}^*$ if we can better minimize (1). Setting $\mu = 0$ would reduce to max-Bleu updating; setting $\mu = \infty$ would never update **w** at all. Setting $\mu = 0.5$ reduces to equation (10).

Figure 1 shows the 10-best unique translations for a single input sentence according to equation (11) under various settings of $\mu$. The points at far right are the translations that are scored highest according to the model. The $\mu = 0$ points in the upper-left corner are typical of oracle translations that would be selected under the max-Bleu policy: they indeed have a very high Bleu score, but are far removed from the translations preferred by the model; thus they would cause violent updates to $\mathbf{w}$. Local updating would select the topmost point labeled $\mu = 1$. Our scheme would select one of the $\mu = 0.5$ points, which have Bleu scores almost as high as the max-Bleu translations, yet are not very far from the translations preferred by the model.

## 2.4 Selecting hypothesis translations

What is the set $\{\mathbf{e}_{ij}\}$ of translation hypotheses? Ideally we would let it be the set of all possible translations, and let the objective function (1) take all of them into account. This is the approach taken by Taskar et al. (2004), but their approach assumes that the loss function can be decomposed into local loss functions. Since our loss function cannot be so decomposed, we select:

- the 10-best translations according to the model;

we then rescore the forest to obtain

- the 10-best translations according to equation (11) with $\mu = 0.5$, the first of which is the oracle translation, and
- the 10-best translations with $\mu = \infty$, to serve as negative examples.

The last case is what Crammer et al. (2006) call *max-loss updating* (where "loss" refers to the generalized hinge loss) and Taskar et al. (2005) call *loss-augmented inference*. The rationale here is that since the objective (1) tries to minimize $\max_j(\ell_{ij} - \Delta\mathbf{h}_{ij} \cdot \mathbf{w}')$, we should include the translations that have the highest $(\ell_{ij} - \Delta\mathbf{h}_{ij} \cdot \mathbf{w})$ in order to approximate the effect of using the whole forest.

See Figure 1 again for an illustration of the hypotheses selected for a single sentence. The max-Bleu points in the upper left are not included (and would have no effect even if they were included). The $\mu = \infty$ points in the lower-right are the negative examples: they are poor translations that are scored too high by the model, and the learning algorithm attempts to shift them to the left.

To perform the forest rescoring, we need to use several approximations, since an exact search for Bleu-optimal translations is NP-hard (Leusch et al., 2008). For every derivation $\mathbf{e}$ in the forest, we calculate a vector $\mathbf{c}(\mathbf{e})$ of counts as in Section 2.2 except using *unclipped* counts of $n$-gram matches (Dreyer et al., 2007), that is, the number of matches for an $n$-gram can be greater than the number of occurrences of the $n$-gram in any reference translation. This can be done efficiently by calculating $\mathbf{c}$ for every hyperedge (rule application) in the forest:

- the number of output words generated by the rule
- the effective reference length scaled by the fraction of the input sentence consumed by the rule
- the number of $n$-grams formed by the application of the rule ($1 \leq n \leq 4$)
- the (unclipped) number of $n$-gram matches formed by the application of the rule ($1 \leq n \leq 4$)

We keep track of $n$-grams using the same scheme used to incorporate an $n$-gram language model into the decoder (Wu, 1996; Chiang, 2007).

To find the best derivation in the forest, we traverse it bottom-up as usual, and for every set of alternative subtranslations, we select the one with the highest score. But here a rough approximation lurks, because we need to calculate $B$ on the nodes of the forest, but $B$ does not have the optimal substructure property, i.e., the optimal score of a parent node cannot necessarily be calculated from the optimal scores of its children. Nevertheless, we find that this rescoring method is good enough for generating high-Bleu oracle translations and low-Bleu negative examples.

## 2.5 Parallelization

One convenient property of MERT is that it is embarrassingly parallel: we decode the entire tuning set sending different sentences to different processors, and during optimization of feature weights, different random restarts can be sent to different processors. In order to make MIRA comparable in efficiency to MERT, we must parallelize it. But with an online learning algorithm, parallelization requires a little more coordination. We run MIRA on each processor simultaneously, with each maintaining its own weight vector. A master process distributes different sentences from the tuning set to each of the processors; when each processor finishes decoding a sentence, it transmits the resulting hypotheses, with their losses, to all the other processors and receives any hypotheses waiting from other processors. Those hypotheses were generated from different weight vectors, but can still provide useful information. The sets of hypotheses thus collected are then processed as one batch. When the whole training process is finished, we simply average all the weight vectors from all the processors.

Having described our training algorithm, which includes several practical improvements to Watanabe et al.'s usage of MIRA, we proceed in the remainder of the paper to demonstrate the utility of the our training algorithm on models with large numbers of structurally sensitive features.

## 3 Soft syntactic constraints

The first features we explore are based on a line of research introduced by Chiang (2005) and improved on by Marton and Resnik (2008). A hierarchical phrase-based translation model is based on synchronous context-free grammar, but does not normally use any syntactic information derived from linguistic knowledge or treebank data: it uses translation rules that span any string of words in the input sentence, without regard for parser-defined syntactic constituency boundaries. Chiang (2005) experimented with a constituency feature that rewarded rules whose source language side exactly spans a syntactic constituent according to the output of an external source-language parser. This feature can be viewed as a *soft syntactic constraint*: it biases the model toward translations that respect syntactic structure, but does not force it to use them. However, this more syntactically aware model, when tested in Chinese-English translation, did not improve translation performance.

Recently, Marton and Resnik (2008) revisited the idea of constituency features, and succeeded in showing that finer-grained soft syntactic constraints yield substantial improvements in BLEU score for both Chinese-English and Arabic-English translation. In addition to adding separate features for different syntactic nonterminals, they introduced a new type of constraint that penalizes rules when the source language side *crosses* the boundaries of a source syntactic constituent, as opposed to simply rewarding rules when they are consistent with the source-language parse tree.

Marton and Resnik optimized their features' weights using MERT. But since MERT does not scale well to large numbers of feature weights, they were forced to test individual features and manually selected feature combinations each in a separate model. Although they showed gains in translation performance for several such models, many larger, potentially better feature combinations remained unexplored. Moreover, the best-performing feature subset was different for the two language pairs, suggesting that this labor-intensive feature selection process would have to be repeated for each new language pair.

Here, we use MIRA to optimize Marton and Resnik's finer-grained single-category features all at once. We define below two sets of features, a coarse-grained class that combines several constituency categories, and a fine-grained class that puts different categories into different features. Both kinds of features were used by Marton and Resnik, but only a few at a time. Crucially, our training algorithm provides the ability to train all the fine-grained features, a total of 34 feature weights, simultaneously.

**Coarse-grained features** As the basis for coarse-grained syntactic features, we selected the following nonterminal labels based on their frequency in the tuning data, whether they frequently cover a span of more than one word, and whether they represent linguistically relevant constituents: NP, PP, S, VP, SBAR, ADJP, ADVP, and QP. We define two new features, one which fires when a rule's source side span in the input sentence matches any of the above-mentioned labels in the input parse, and another which fires when a rule's source side span crosses a boundary of one of these labels (e.g., its source side span only partially covers the words in a VP subtree, and it also covers some or all or the words outside the VP subtree). These two features are equivalent to Marton and Resnik's $\text{XP}^{=}$ and $\text{XP}^{+}$ feature combinations, respectively.

**Fine-grained features** We selected the following nonterminal labels that appear more than 100 times in the tuning data: NP, PP, S, VP, SBAR, ADJP, WHNP, PRT, ADVP, PRN, and QP. The labels that were excluded were parts of speech, nonconstituent labels like FRAG, or labels that occurred only two or three times. For each of these labels $X$, we added a separate feature that fires when a rule's source side span in the input sentence matches $X$, and a second feature that fires when a span crosses a boundary of $X$. These features are similar to Marton and Resnik's $X^{=}$ and $X^{+}$, except that our set includes features for WHNP, PRT, and PRN.

## 4 Structural distortion features

In addition to parser-based syntactic constraints, which were introduced in prior work, we introduce a completely new set of features aimed at improving the modeling of reordering within Hiero. Again, the feature definition gives rise to a larger number of features than one would expect to train successfully using MERT.

In a phrase-based model, reordering is performed both within phrase pairs and by the phrase-reordering model. Both mechanisms are able to learn that longer-distance reorderings are more costly than shorter-distance reorderings: phrase pairs, because phrases that involve more extreme reorderings will (presumably) have a lower count in the data, and phrase reordering, because models are usually explicitly dependent on distance.

By contrast, in a hierarchical model, all reordering is performed by a single mechanism, the rules of the grammar. In some cases, the model will be able to learn a preference for shorter-distance reorderings, as in a phrase-based system, but in the case of a word being reordered across a nonterminal, or two nonterminals being reordered, there is no dependence in the model on the size of the nonterminal or nonterminals involved in reordering.

So, for example, if we have rules

$$\mathrm{X} \rightarrow (\text{il dit } \mathrm{X}_1, \text{he said } \mathrm{X}_1) \qquad (12)$$

$$\mathrm{X} \rightarrow (\text{il dit } \mathrm{X}_1, \mathrm{X}_1 \text{ he said}) \qquad (13)$$

we might expect that rule (12) is more common in general, but that rule (13) becomes more and more

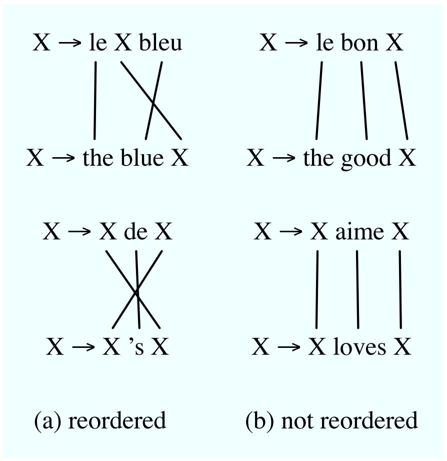

Figure 2: Classifying nonterminal occurrences for the structural distortion model.

rare as $\mathrm{X}_1$ gets larger. The default Hiero features have no way to learn this.

To address this defect, we can classify every nonterminal pair occurring on the right-hand side of each grammar rule as "reordered" or "not reordered", that is, whether it intersects any other word alignment link or nonterminal pair (see Figure 2). We then define coarse- and fine-grained versions of the structural distortion model.

**Coarse-grained features** Let $R$ be a binary-valued random variable that indicates whether a nonterminal occurrence is reordered, and let $S$ be an integer-valued random variable that indicates how many source words are spanned by the nonterminal occurrence. We can estimate $P(R \mid S)$ via relative-frequency estimation from the rules as they are extracted from the parallel text, and incorporate this probability as a new feature of the model.

**Fine-grained features** A difficulty with the coarse-grained reordering features is that the grammar extraction process finds overlapping rules in the training data and might not give a sensible probability estimate; moreover, reordering statistics from the training data might not carry over perfectly into the translation task (in particular, the training data may have some very freely-reordering translations that one might want to avoid replicating in translation). As an alternative, we introduce a fine-grained version of our distortion model that can be trained directly in the translation task as follows: define a separate binary feature for each value of $(R, S)$, where $R$ is as above and $S \in \{\star, 1, \ldots, 9, \geq 10\}$ and $\star$ means any size. For example, if a nonterminal with span 11 has its contents reordered, then the features (true, $\geq$10) and (true, $\star$) would both fire. Grouping all sizes of 10 or more into a single feature is designed to avoid overfitting.

Again, using MIRA makes it practical to train with the full fine-grained feature set—coincidentally also a total of 34 features.

## 5 Experiment and results

We now describe our experiments to test MIRA and our features, the soft-syntactic constraints and the structural distortion features, on an Arabic-English translation task. It is worth noting that this experimentation is on a larger scale than Watanabe et al.'s (2007), and considerably larger than Marton and Resnik's (2008).

### 5.1 Experimental setup

The baseline model was Hiero with the following baseline features (Chiang, 2005; Chiang, 2007):

- two language models
- phrase translation probabilities $p(f \mid e)$ and $p(e \mid f)$
- lexical weighting in both directions (Koehn et al., 2003)
- word penalty
- penalties for:
  - automatically extracted rules
  - identity rules (translating a word into itself)
  - two classes of number/name translation rules
  - glue rules

The probability features are base-100 log-probabilities.

The rules were extracted from all the allowable parallel text from the NIST 2008 evaluation (152+175 million words of Arabic+English), aligned by IBM Model 4 using GIZA++ (union of both directions). Hierarchical rules were extracted from the most in-domain corpora (4.2+5.4 million words) and phrases were extracted from the remainder. We trained the coarse-grained distortion model on 10,000 sentences of the training data.

Two language models were trained, one on data similar to the English side of the parallel text and one on 2 billion words of English. Both were 5-gram models with modified Kneser-Ney smoothing, lossily compressed using a perfect-hashing scheme similar to that of Talbot and Brants (2008) but using minimal perfect hashing (Botelho et al., 2005).

We partitioned the documents of the NIST 2004 (newswire) and 2005 Arabic-English evaluation data into a tuning set (1178 sentences) and a development set (1298 sentences). The test data was the NIST 2006 Arabic-English evaluation data (NIST part, newswire and newsgroups, 1529 sentences).

To obtain syntactic parses for this data, we tokenized it according to the Arabic Treebank standard using AMIRA (Diab et al., 2004), parsed it with the Stanford parser (Klein and Manning, 2003), and then forced the trees back into the MT system's tokenization.[1]

We ran both MERT and MIRA on the tuning set using 20 parallel processors. We stopped MERT when the score on the tuning set stopped increasing, as is common practice, and for MIRA, we used the development set to decide when to stop training.[2] In our runs, MERT took an average of 9 passes through the tuning set and MIRA took an average of 8 passes. (For comparison, Watanabe et al. report decoding their tuning data of 663 sentences 80 times.)

### 5.2 Results

Table 1 shows the results of our experiments with the training methods and features described above. All significance testing was performed against the first line (MERT baseline) using paired bootstrap resampling (Koehn, 2004).

First of all, we find that MIRA is competitive with MERT when both use the baseline feature set. In-

[1]The only notable consequence this had for our experimentation is that proclitic Arabic prepositions were fused onto the first word of their NP object, so that the PP and NP brackets were coextensive.

[2]We chose this policy for MIRA to avoid overfitting. However, we could have used the tuning set for this purpose, just as with MERT: in none of our runs would this change have made more than a 0.2 Bleu difference on the development set.

| | | | Dev | NIST 06 (NIST part) | | |
|---|---|---|---|---|---|---|
| Train | Features | # | nw | nw | ng | nw+ng |
| MERT | baseline | 12 | 52.0 | 50.5 | 32.4 | 44.6 |
| | syntax (coarse) | 14 | 52.2 | 50.9 | $33.0^{+}$ | $45.0^{+}$ |
| | syntax (fine) | 34 | 52.1 | 50.4 | $33.5^{++}$ | 44.8 |
| | distortion (coarse) | 13 | 52.3 | $51.3^{+}$ | $34.3^{++}$ | $45.8^{++}$ |
| | distortion (fine) | 34 | 52.0 | 50.9 | $34.5^{++}$ | $45.5^{++}$ |
| MIRA | baseline | 12 | 52.0 | $49.8^{-}$ | $34.2^{++}$ | $45.3^{++}$ |
| | syntax (fine) | 34 | $53.1^{++}$ | $51.3^{+}$ | $34.5^{++}$ | $46.4^{++}$ |
| | distortion (fine) | 34 | $53.3^{++}$ | $51.5^{++}$ | $34.7^{++}$ | $46.7^{++}$ |
| | distortion+syntax (fine) | 56 | $53.6^{++}$ | $52.0^{++}$ | $35.0^{++}$ | $47.2^{++}$ |

Table 1: Comparison of MERT and MIRA on various feature sets. Key: # = number of features; nw = newswire, ng = newsgroups; + or ++ = significantly better than MERT baseline ($p < 0.05$ or $p < 0.01$, respectively), − = significantly worse than MERT baseline ($p < 0.05$).

deed, the MIRA system scores significantly higher on the test set; but if we break the test set down by genre, we see that the MIRA system does slightly worse on newswire and better on newsgroups. (This is largely attributable to the fact that the MIRA translations tend to be longer than the MERT translations, and the newsgroup references are also relatively longer than the newswire references.)

When we add more features to the model, the two training methods diverge more sharply. When training with MERT, the coarse-grained pair of syntax features yields a small improvement, but the fine-grained syntax features do not yield any further improvement. By contrast, when the fine-grained features are trained using MIRA, they yield substantial improvements. We observe similar behavior for the structural distortion features: MERT is not able to take advantage of the finer-grained features, but MIRA is. Finally, using MIRA to combine both classes of features, 56 in all, produces the largest improvement, 2.6 Bleu points over the MERT baseline on the full test set.

We also tested some of the differences between our training method and Watanabe et al.'s (2007); the results are shown in Table 2. Compared with local updating (line 2), our method of selecting the oracle translation and negative examples does better by 0.5 Bleu points on the development data. Using loss-augmented inference to add negative examples to local updating (line 3) does not appear to help. Nevertheless, the negative examples are important: for if we use our method for selecting the oracle translation without the additional negative examples (line 4), the algorithm fails, generating very long translations and unable to find a weight setting to shorten them. It appears, then, that the additional negative examples enable the algorithm to reliably learn from the enhanced oracle translations.

| Setting | Dev |
|---|---|
| full | 53.6 |
| local updating, no LAI | $53.1^{-}$ |
| local updating, LAI | $53.0^{--}$ |
| $\mu = 0.5$ oracle, no LAI | failed |
| no sharing of updates | $53.1^{--}$ |

Table 2: Effect of removing various improvements in learning method. Key: − or −− = significantly worse than full system ($p < 0.05$ or $p < 0.01$, respectively); LAI = loss-augmented inference for additional negative examples.

Finally, we compared our parallelization method against a simpler method in which all processors learn independently and their weight vectors are all averaged together (line 5). We see that sharing information among the processors makes a significant difference.

## 6 Conclusions

In this paper, we have brought together two existing lines of work: the training method of Watanabe et al. (2007), and the models of Chiang (2005) and Marton and Resnik (2008). Watanabe et al.'s work showed that large-margin training with MIRA can be made feasible for state-of-the-art MT systems by using a manageable tuning set; we have demonstrated that parallel processing and exploiting more of the parse forest improves MIRA's performance and that, even using the same set of features, MIRA's performance compares favorably to MERT in terms of both translation quality and computational cost.

Marton and Resnik (2008) showed that it is possible to improve translation in a data-driven framework by incorporating source-side syntactic analysis in the form of soft syntactic constraints. This work joins a growing body of work demonstrating the utility of syntactic information in statistical MT. In the area of source-side syntax, recent research has continued to improve tree-to-string translation models, soften the constraints of the input tree in various ways (Mi et al., 2008; Zhang et al., 2008), and extend phrase-based translation with source-side soft syntactic constraints (Cherry, 2008). All this work shows strong promise, but Marton and Resnik's soft syntactic constraint approach is particularly appealing because it can be used unobtrusively with any hierarchically-structured translation model. Here, we have shown that using MIRA to weight all the constraints at once removes the crucial drawback of the approach, the problem of feature selection.

Finally, we have introduced novel structural distortion features to fill a notable gap in the hierarchical phrase-based approach. By capturing how reordering depends on constituent length, these features improve translation quality significantly. In sum, we have shown that removing the bottleneck of MERT opens the door to many possibilities for better translation.

## Acknowledgments

Thanks to Michael Bloodgood for performing initial simulations of parallelized perceptron training. Thanks also to John DeNero, Kevin Knight, Daniel Marcu, and Fei Sha for valuable discussions and suggestions. This research was supported in part by DARPA contract HR0011-06-C-0022 under subcontract to BBN Technologies and HR0011-06-02-001 under subcontract to IBM.

## References

Abhishek Arun and Philipp Koehn. 2007. Online learning methods for discriminative training of phrase based statistical machine translation. In *Proc. MT Summit XI*.

Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proc. ACL-08: HLT*.

Fabiano C. Botelho, Yoshiharu Kohayakawa, and Nivio Ziviani. 2005. A practical minimal perfect hashing method. In *4th International Workshop on Efficient and Experimental Algorithms (WEA05)*.

Daniel Cer, Daniel Jurafsky, and Christopher D. Manning. 2008. Regularization and search for minimum error rate training. In *Proc. Third Workshop on Statistical Machine Translation*.

Colin Cherry. 2008. Cohesive phrase-based decoding for statistical machine translation. In *Proc. ACL-08: HLT*.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. ACL 2005*.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2).

Michael Collins. 2002. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proc. EMNLP 2002*.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic tagging of Arabic text: From raw text to base phrase chunks. In *Proc. HLT/NAACL 2004*. Companion volume.

Markus Dreyer, Keith Hall, and Sanjeev Khudanpur. 2007. Comparing reordering constraints for SMT using efficient Bleu oracle computation. In *Proc. 2007 Workshop on Syntax and Structure in Statistical Translation*.

Kevin Duh and Katrin Kirchoff. 2008. Beyond log-linear models: Boosted minimum error rate training for n-best re-ranking. In *Proc. ACL-08: HLT, Short Papers*.

Yoav Freund and Robert E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296.

Dan Klein and Chris D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT-NAACL 2003*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP 2004*.

Gregor Leusch, Evgeny Matusov, and Hermann Ney. 2008. Complexity of finding the BLEU-optimal hypothesis in a confusion network. In *Proc. EMNLP 2008*. This volume.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proc. COLING-ACL 2006*.

Chin-Yew Lin and Franz Josef Och. 2004. ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *Proc. COLING 2004*.

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proc. ACL-08: HLT*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proc. ACL 2005*.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proc. ACL-08: HLT*.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. ACL 2002*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL 2003*.

John C. Platt. 1998. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 195–208. MIT Press.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proc. COLING/ACL 2006, Poster Sessions*.

David Talbot and Thorsten Brants. 2008. Randomized language models via perfect hash functions. In *Proc. ACL-08: HLT*.

Ben Taskar, Dan Klein, Michael Collins, Daphne Koller, and Christopher Manning. 2004. Max-margin parsing. In *Proc. EMNLP 2004*, pages 1–8.

Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. 2005. Learning structured prediction models: A large margin approach. In *Proc. ICML 2005*.

Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical MT. In *Proc. COLING-ACL 2006*.

Joseph Turian, Benjamin Wellington, and I. Dan Melamed. 2007. Scalable discriminative learning for natural language parsing and translation. In *Advances in Neural Information Processing Systems 19 (NIPS 2006)*.

Taro Watanabe, Jun Suzuki, Hajime Tsukuda, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proc. EMNLP 2007*.

Dekai Wu. 1996. A polynomial-time algorithm for statistical machine translation. In *Proc. 34th Annual Meeting of the Association for Computational Linguistics*, pages 152–158.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proc. ACL-08: HLT*.