11,001 New Features for Statistical Machine Translation

David Chiang and Kevin Knight
USC Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292 USA

Wei Wang
Language Weaver, Inc.
4640 Admiralty Way, Suite 1210
Marina del Rey, CA 90292 USA

Abstract
We use the Margin Infused Relaxed Algorithm of Crammer et al. to add a large number of new features to two machine translation systems: the Hiero hierarchical phrase-based translation system and our syntax-based translation system. On a large-scale Chinese-English translation task, we obtain statistically significant improvements of +1.5 BLEU and +1.1 BLEU, respectively. We analyze the impact of the new features and the performance of the learning algorithm.

1 Introduction
What linguistic features can improve statistical machine translation (MT)? This is a fundamental question for the discipline, particularly as it pertains to improving the best systems we have. Further:

- Do syntax-based translation systems have unique and effective levers to pull when designing new features?
- Can large numbers of feature weights be learned efficiently and stably on modest amounts of data?

In this paper, we address these questions by experimenting with a large number of new features. We add more than 250 features to improve a syntax-based MT system—already the highest-scoring single system in the NIST 2008 Chinese-English common-data track—by +1.1 BLEU. We also add more than 10,000 features to Hiero (Chiang, 2005) and obtain a +1.5 BLEU improvement.

Many of the new features use syntactic information, and in particular depend on information that is available only inside a syntax-based translation model. Thus they widen the advantage that syntax-based models have over other types of models.

The models are trained using the Margin Infused Relaxed Algorithm or MIRA (Crammer et al., 2006) instead of the standard minimum-error-rate training or MERT algorithm (Och, 2003). Our results add to a growing body of evidence (Watanabe et al., 2007; Chiang et al., 2008) that MIRA is preferable to MERT across languages and systems, even for very large-scale tasks.

2 Related Work
The work of Och et al. (2004) is perhaps the best-known study of new features and their impact on translation quality. However, it had a few shortcomings. First, it used the features for reranking n-best lists of translations, rather than for decoding or forest reranking (Huang, 2008). Second, it attempted to incorporate syntax by applying off-the-shelf part-of-speech taggers and parsers to MT output, a task these tools were never designed for. By contrast, we incorporate features directly into hierarchical and syntax-based decoders.

A third difficulty with Och et al.’s study was that it used MERT, which is not an ideal vehicle for feature exploration because it is observed not to perform well with large feature sets. Others have introduced alternative discriminative training methods (Tillmann and Zhang, 2006; Liang et al., 2006; Turian et al., 2007; Blunsom et al., 2008; Macherey et al., 2008), in which a recurring challenge is scalability: to train many features, we need many train-
ing examples, and to train discriminatively, we need to search through all possible translations of each training example. Another line of research (Watanabe et al., 2007; Chiang et al., 2008) tries to squeeze as many features as possible from a relatively small dataset. We follow this approach here.

3 Systems Used

3.1 Hiero
Hiero (Chiang, 2005) is a hierarchical, string-to-string translation system. Its rules, which are extracted from unparsed, word-aligned parallel text, are synchronous CFG productions, for example:

\[ X \rightarrow X_1 \text{ de } X_2, X_2 \text{ of } X_1 \]

As the number of nonterminals is limited to two, the grammar is equivalent to an inversion transduction grammar (Wu, 1997).

The baseline model includes 12 features whose weights are optimized using MERT. Two of the features are \( n \)-gram language models, which require intersecting the synchronous CFG with finite-state automata representing the language models. This grammar can be parsed efficiently using cube pruning (Chiang, 2007).

3.2 Syntax-based system
Our syntax-based system transforms source Chinese strings into target English syntax trees. Following previous work in statistical MT (Brown et al., 1993), we envision a noisy-channel model in which a language model generates English, and then a translation model transforms English trees into Chinese. We represent the translation model as a tree transducer (Knight and Graehl, 2005). It is obtained from bilingual text that has been word-aligned and whose English side has been syntactically parsed. From this data, we use the the GHKM minimal-rule extraction algorithm of (Galley et al., 2004) to yield rules like:

\[ \text{NP-C}(x_0;\text{NPB PP(IN(of } x_1;\text{NPB})) \leftrightarrow x_1 \text{ de } x_0 \]

Though this rule can be used in either direction, here we use it right-to-left (Chinese to English). We follow Galley et al. (2006) in allowing unaligned Chinese words to participate in multiple translation rules, and in collecting larger rules composed of minimal rules. These larger rules have been shown to substantially improve translation accuracy (Galley et al., 2006; DeNeefe et al., 2007).

We apply Good-Turing discounting to the transducer rule counts and obtain probability estimates:

\[ P(\text{rule}) = \frac{\text{count}(\text{rule})}{\text{count}(\text{LHS-root}(\text{rule}))} \]

When we apply these probabilities to derive an English sentence \( e \) and a corresponding Chinese sentence \( c \), we wind up with the joint probability \( P(e, c) \).

The baseline model includes \( \log P(e, c) \), the two \( n \)-gram language models \( \log P(e) \), and other features for a total of 25. For example, there is a pair of features to punish rules that drop Chinese content words or introduce spurious English content words. All features are linearly combined and their weights are optimized using MERT.

For efficient decoding with integrated \( n \)-gram language models, all transducer rules must be binarized into rules that contain at most two variables and can be incrementally scored by the language model (Zhang et al., 2006). Then we use a CKY-style parser (Yamada and Knight, 2002; Galley et al., 2006) with cube pruning to decode new sentences.

We include two other techniques in our baseline. To get more general translation rules, we restructure our English training trees using expectation-maximization (Wang et al., 2007), and to get more specific translation rules, we relabel the trees with up to 4 specialized versions of each nonterminal symbol, again using expectation-maximization and the split/merge technique of Petrov et al. (2006).

3.3 MIRA training
We incorporate all our new features into a linear model (Och and Ney, 2002) and train them using MIRA (Crammer et al., 2006), following previous work (Watanabe et al., 2007; Chiang et al., 2008).

Let \( e \) stand for output strings or their derivations, and let \( \mathbf{h}(e) \) stand for the feature vector for \( e \). Initialize the feature weights \( w \). Then, repeatedly:

- Select a batch of input sentences \( f_1, \ldots, f_m \) and decode each \( f_i \) to obtain a forest of translations.
- For each \( i \), select from the forest a set of hypothesis translations \( e_{i1}, \ldots, e_{im} \), which are the
10-best translations according to each of:

\[ h(e) \cdot w \]
\[ \text{BLEU}(e) + h(e) \cdot w \]
\[ -\text{BLEU}(e) + h(e) \cdot w \]

(1)

- For each \( i \), select an oracle translation:

\[ e^* = \arg \max_e (\text{BLEU}(e) + h(e) \cdot w) \]

(2)

Let \( \Delta h_{ij} = h(e^*_j) - h(e_{ij}). \)

- For each \( e_{ij} \), compute the loss

\[ \ell_{ij} = \text{BLEU}(e^*_j) - \text{BLEU}(e_{ij}) \]

(3)

- Update \( w \) to the value of \( w' \) that minimizes:

\[
\frac{1}{2} \| w' - w \|^2 + C \sum_{i=1}^{m} \max_{1 \leq j \leq n} (\ell_{ij} - \Delta h_{ij} \cdot w')
\]

(4)

where \( C = 0.01 \). This minimization is performed by a variant of sequential minimal optimization (Platt, 1998).

Following Chiang et al. (2008), we calculate the sentence \text{BLEU} scores in (1), (2), and (3) in the context of some previous 1-best translations. We run 20 of these learners in parallel, and when training is finished, the weight vectors from all iterations of all learners are averaged together.

Since the interface between the trainer and the decoder is fairly simple—for each sentence, the decoder sends the trainer a forest, and the trainer returns a weight update—it is easy to use this algorithm with a variety of CKY-based decoders: here, we are using it in conjunction with both the Hiero decoder and our syntax-based decoder.

4 Features

In this section, we describe the new features introduced on top of our baseline systems.

Discount features Both of our systems calculate several features based on observed counts of rules in the training data. Though the syntax-based system uses Good-Turing discounting when computing the \( P(e, c) \) feature, we find, as noted above, that it uses quite a few one-count rules, suggesting that their probabilities have been overestimated. We can directly attack this problem by adding features \( \text{count}_i \) that reward or punish rules seen \( i \) times, or features \( \text{count}_{[i,j]} \) for rules seen between \( i \) and \( j \) times.

4.1 Target-side features

String-to-tree MT offers some unique levers to pull, in terms of target-side features. Because the system outputs English trees, we can analyze output trees on the tuning set and design new features to encourage the decoder to produce more grammatical trees.

Rule overlap features While individual rules observed in decoder output are often quite reasonable, two adjacent rules can create problems. For example, a rule that has a variable of type IN (preposition) needs another rule rooted with IN to fill the position. If the second rule supplies the wrong preposition, a bad translation results. The IN node here is an overlap point between rules. Considering that certain nonterminal symbols may be more reliable overlap points than others, we create a binary feature for each nonterminal. A rule like:

\[ \text{IN(at)} \leftrightarrow \text{zai} \]

will have feature \text{rule-root-IN} set to 1 and all other rule-root features set to 0. Our rule root features range over the original (non-split) nonterminal set; we have 105 in total. Even though the rule root features are locally attached to individual rules—and therefore cause no additional problems for the decoder search—they are aimed at problematic rule/rule interactions.

Bad single-level rewrites Sometimes the decoder uses questionable rules, for example:

\[ \text{PP}(x_0:\text{VBN} x_1:\text{NP-C}) \leftrightarrow x_0 x_1 \]

This rule is learned from 62 cases in our training data, where the VBN is almost always the word \textit{given}. However, the decoder misuses this rule with other VBNs. So we can add a feature that penalizes any rule in which a PP dominates a VBN and NP-C. The feature class \text{bad-rewrite} comprises penalties for the following configurations based on our analysis of the tuning set:

\[
\begin{align*}
\text{PP} & \rightarrow \text{VBN NP-C} \\
\text{PP-BAR} & \rightarrow \text{NP-C IN} \\
\text{VP} & \rightarrow \text{NP-C PP} \\
\text{CONJP} & \rightarrow \text{RB IN}
\end{align*}
\]
Node count features It is possible that the decoder creates English trees with too many or too few nodes of a particular syntactic category. For example, there may be a tendency to generate too many determiners or past-tense verbs. We therefore add a count feature for each of the 109 (non-split) English nonterminal symbols. For a rule like

\[
\text{NPB}(\text{NNP}(\text{us}) \text{ NNP}(\text{president}) \, x_0 : \text{NNP}) \leftrightarrow \text{meiguo zongtong} \, x_0
\]

the feature \text{node-count-NPB} gets value 1, \text{node-count-NNP} gets value 2, and all others get 0.

Insertion features Among the rules we extract from bilingual corpora are target-language insertion rules, which have a word on the English side, but no words on the source Chinese side. Sample syntax-based insertion rules are:

\[
\text{NPB}(\text{DT}(\text{the}) \, x_0 : \text{NN}) \leftrightarrow x_0
\]

\[
S(x_0 : \text{NP-C VP}(\text{VBZ}(\text{is}) \, x_1 : \text{VP-C})) \leftrightarrow x_0 \, x_1
\]

We notice that our decoder, however, frequently fails to insert words like \textit{is} and \textit{are}, which often have no equivalent in the Chinese source. We also notice that the-insertion rules sometimes have a good effect, as in the translation “in the bloom of youth,” but other times have a bad effect, as in “people seek areas of the conspiracy.”

Each time the decoder uses (or fails to use) an insertion rule, it incurs some risk. There is no guarantee that the interaction of the rule probabilities and the language model provides the best way to manage this risk. We therefore provide MIRA with a feature for each of the most common English words appearing in insertion rules, e.g., \text{insert-the} and \text{insert-is}. There are 35 such features.

4.2 Source-side features

We now turn to features that make use of source-side context. Although these features capture dependencies that cross boundaries between rules, they are still local in the sense that no new states need to be added to the decoder. This is because the entire source sentence, being fixed, is always available to every feature.

Soft syntactic constraints Neither of our systems uses source-side syntactic information; hence, both could potentially benefit from soft syntactic constraints as described by Marton and Resnik (2008). In brief, these features use the output of an independent syntactic parser on the source sentence, rewarding decoder constituents that match syntactic constituents and punishing decoder constituents that cross syntactic constituents. We use separately-tunable features for each syntactic category.

Structural distortion features Both of our systems have rules with variables that generalize over possible fillers, but neither system’s basic model conditions a rule application on the size of a filler, making it difficult to distinguish long-distance reorderings from short-distance reorderings. To remedy this problem, Chiang et al. (2008) introduce a structural distortion model, which we include in our experiment. Our syntax-based baseline includes the generative version of this model already.

Word context During rule extraction, we retain word alignments from the training data in the extracted rules. (If a rule is observed with more than one set of word alignments, we keep only the most frequent one.) We then define, for each triple \((f, e, f_{+1})\), a feature that counts the number of times that \(f\) is aligned to \(e\) and \(f_{+1}\) occurs to the right of \(f\); and similarly for triples \((f, e, f_{-1})\) with \(f_{-1}\) occurring to the left of \(f\). In order to limit the size of the model, we restrict words to be among the 100 most frequently occurring words from the training data; all other words are replaced with a token \text{<unk>}

These features are somewhat similar to features used by Watanabe et al. (2007), but more in the spirit of features used in the word sense disambiguation model introduced by Lee and Ng (2002) and incorporated as a submodel of a translation system by Chan et al. (2007); here, we are incorporating some of its features directly into the translation model.

5 Experiments

For our experiments, we used a 260 million word Chinese/English bitext. We ran GIZA++ on the entire bitext to produce IBM Model 4 word alignments, and then the link deletion algorithm (Fossum et al., 2008) to yield better-quality alignments. For
Table 1: Adding new features with MIRA significantly improves translation accuracy. Scores are case-insensitive IBM BLEU scores. * or ** = significantly better than MERT baseline (p < 0.05 or 0.01, respectively).

Table 2: Weights learned for discount features. Negative weights indicate bonuses; positive weights indicate penalties.

The results (Table 1) show significant improvements in both systems (p < 0.01) over already very strong MERT baselines. Adding the source-side and discount features to Hiero yields a +1.5 BLEU improvement, and adding the target-side syntax and discount features to the syntax-based system yields a +1.1 BLEU improvement. The results also show that for Hiero, the various classes of features contributed roughly equally; for the syntax-based system, we see that two of the feature classes make small contributions but time constraints unfortunately did not permit isolated testing of all feature classes.

6 Analysis

How did the various new features improve the translation quality of our two systems? We begin by examining the discount features. For these features, we used slightly different schemes for the two systems, shown in Table 2 with their learned feature weights. We see in both cases that one-count rules are strongly penalized, as expected.
Reward
−0.42 a +0.67 of
−0.13 are +0.56 the
−0.09 at +0.47 comma
−0.09 on +0.13 period
−0.05 was +0.11 in
−0.05 from +0.08 for
−0.04 's +0.06 to
−0.04 by +0.05 will
−0.04 is +0.04 and
−0.03 it +0.02 as
−0.03 its +0.02 have

Table 3: Weights learned for inserting target English words with rules that lack Chinese words.

6.1 Syntax features

Table 3 shows word-insertion feature weights. The system rewards insertion of forms of be; examples 1–3 in Figure 1 show typical improved translations that result. Among determiners, inserting a is rewarded, while inserting the is punished. This seems to be because the is often part of a fixed phrase, such as the White House, and therefore comes naturally as part of larger phrasal rules. Inserting the outside these fixed phrases is a risk that the generative model is too inclined to take. We also note that the system learns to punish unmotivated insertions of commas and periods, which get into our grammar via quirks in the MT training data.

Table 4 shows weights for rule-overlap features. MIRA punishes the case where rules overlap with an IN (preposition) node. This makes sense: if a rule has a variable that can be filled by any English preposition, there is a risk that an incorrect preposition will fill it. On the other hand, splitting at a period is a safe bet, and frees the model to use rules that dig deeper into NP and VP trees when constructing a top-level S. Table 5 shows weights for generated English nonterminals: SBAR-C nodes are rewarded and commas are punished.

The combined effect of all weights is subtle. To interpret them further, it helps to look at gross changes in the system’s behavior. For example, a major error in the baseline system is to move “X said” or “X asked” from the beginning of the Chinese input to the middle or end of the English trans-
lation. The error occurs with many speaking verbs, and each time, we trace it to a different rule. The problematic rules can even be non-lexical, e.g.:

\[ S(x_0:\text{NP-C} x_1:\text{VP} x_2:\text{NP-C} x_3:\text{NP-C} x_4:\text{VP} x_5:) \]
\[ \leftrightarrow x_3 x_4 x_2 x_0 x_1 x_5 \]

It is therefore difficult to come up with a straightforward feature to address the problem. However, when we apply MIRA with the features already listed, these translation errors all disappear, as demonstrated by examples 4–5 in Figure 1. Why does this happen? It turns out that in translation hypotheses that move “X said” or “X asked” away from the beginning of the sentence, more commas appear, and fewer S-C and SBAR-C nodes appear. Therefore, the new features work to discourage these hypotheses. Example 6 shows additionally that commas next to speaking verbs are now correctly deleted.

Examples 7–8 in Figure 1 show other kinds of unanticipated improvements. We do not have space for a fuller analysis, but we note that the specific effects we describe above account for only part of the overall \text{BLEU} improvement.

6.2 Word context features

In Table 6 are shown feature weights learned for the word-context features. A surprising number of the highest-weighted features have to do with translations of dates and bylines. Many of the penalties seem to discourage spurious insertion or deletion of frequent words (\textit{for}, ‘s, \textit{said}, parentheses, and quotes). Finally, we note that several of the features (the third- and eighth-ranked reward and twelfth-ranked penalty) shape the translation of \textit{shuo} ‘said’, preferring translations with an overt complementizer \textit{that} and without a comma. Thus these features work together to attack a frequent problem that our target-syntax features also addressed.

Figure 2 shows the performance of Hiero with all of its features on the tuning and test sets over time. The scores on the tuning set rise rapidly, and the scores on the test set also rise, but much more slowly, and there appears to be slight degradation after the 18th pass through the tuning data. This seems in line with the finding of Watanabe et al. (2007) that with on the order of 10,000 features, overfitting is possible, but we can still improve accuracy on new data.

Figure 2: Using over 10,000 word-context features leads to overfitting, but its detrimental effects are modest. Scores on the tuning set were obtained from the 1-best output of the online learning algorithm, whereas scores on the test set were obtained using averaged weights.

Early stopping would have given +0.2 \text{BLEU} over the results reported in Table 1.\footnote{It was this iteration, in fact, which was used to derive the combined feature count used in the title of this paper.}

7 Conclusion

We have described a variety of features for statistical machine translation and applied them to syntax-based and hierarchical systems. We saw that these features, discriminatively trained using MIRA, led to significant improvements, and took a closer look at the results to see how the new features qualitatively improved translation quality. We draw three conclusions from this study.

First, we have shown that these new features can improve the performance even of top-scoring MT systems. Second, these results add to a growing body of evidence that MIRA is preferable to MERT for discriminative training. When training over 10,000 features on a modest amount of data, we, like Watanabe et al. (2007), did observe overfitting, yet saw improvements on new data. Third, we have shown that syntax-based machine translation offers possibilities for features not available in other models, making syntax-based MT and MIRA an especially strong combination for future work.
1 MERT: the united states pending israeli clarification on golan settlement plan
MIRA: the united states is waiting for israeli clarification on golan settlement plan

2 MERT: . . . the average life expectancy of only 18 months , canada 's minority goverment will . . .
MIRA: . . . the average life expectancy of canada’s previous minority government is only 18 months . . .

3 MERT: . . . since un inspectors expelled by north korea . . .
MIRA: . . . since un inspectors were expelled by north korea . . .

4 MERT: another thing is . . . , " he said , " obviously , the first thing we need to do . . .
MIRA: he said : " obviously , the first thing we need to do . . . , and another thing is . . . ."

5 MERT: the actual timing . . . reopened in january , yoon said .
MIRA: yoon said the issue of the timing . . .

6 MERT: . . . us - led coalition forces , said today that the crash . . .
MIRA: . . . us - led coalition forces said today that a us military . . .

7 MERT: . . . and others will feel the danger .
MIRA: . . . and others will not feel the danger .

8 MERT: in residential or public activities within 200 meters of the region . . .
MIRA: within 200 m of residential or public activities area . . .

Figure 1: Improved syntax-based translations due to MIRA-trained weights.

| Bonus | Penalty |
|-------|---------|
| $f$  | $e$  | $context$ | $f$  | $e$  | $context$ |
| -1.19 | <unk> | <unk> | $f_{-1} = \text{ri ‘day’}$ | +1.12 | <unk> | ) |
| -1.01 | <unk> | <unk> | $f_{-1} = ( $ | +0.83 | jiang ‘shall’ | be |
| -0.84 | , | that | $f_{-1} = \text{shuo ‘say’}$ | +0.83 | zhengfu ‘government’ | the |
| -0.82 | yue ‘month’ | <unk> | $f_{+1} = <unk>$ | +0.73 | <unk> | ) |
| -0.78 | " | " | $f_{-1} = <unk>$ | +0.73 | <unk> | ( |
| -0.76 | " | " | $f_{+1} = <unk>$ | +0.72 | <unk> | ) |
| -0.66 | <unk> | <unk> | $f_{+1} = \text{nian ‘year’}$ | +0.70 | <unk> | ( |
| -0.65 | , | that | $f_{+1} = <unk>$ | +0.69 | <unk> | ) |
| ... | | | | +0.66 | <unk> | for |
| | | | | +0.66 | <unk> | ’s |
| | | | | +0.65 | <unk> | said |
| | | | | +0.60 | <unk> | , |

Table 6: Weights learned for word-context features, which fire when English word $e$ is generated aligned to Chinese word $f$, with Chinese word $f_{-1}$ to the left or $f_{+1}$ to the right. Glosses for Chinese words are not part of features.
