A Smorgasbord of Features for Statistical Machine Translation

Franz Josef Och USC/ISI
Daniel Gildea U. of Rochester
Sanjeev Khudanpur Johns Hopkins U.
Anoop Sarkar Simon Fraser U.

Kenji Yamada Xerox/XRCE
Alex Fraser USC/ISI
Shankar Kumar Johns Hopkins U.
Libin Shen U. of Pennsylvania
David Smith Johns Hopkins U.

Katherine Eng Stanford U.
Viren Jain U. of Pennsylvania
Zhen Jin Mt. Holyoke
Dragomir Radev U. of Michigan

Abstract

We describe a methodology for rapid experimentation in statistical machine translation which we use to add a large number of features to a baseline system exploiting features from a wide range of levels of syntactic representation. Feature values were combined in a log-linear model to select the highest scoring candidate translation from an n-best list. Feature weights were optimized directly against the BLEU evaluation metric on held-out data. We present results for a small selection of features at each level of syntactic representation.

1 Introduction

Despite the enormous progress in machine translation (MT) due to the use of statistical techniques in recent years, state-of-the-art statistical systems often produce translations with obvious errors. Grammatical errors include lack of a main verb, wrong word order, and wrong choice of function words. Frequent problems of a less grammatical nature include missing content words and incorrect punctuation.

In this paper, we attempt to address these problems by exploring a variety of new features for scoring candidate translations. A high-quality statistical translation system is our baseline, and we add new features to the existing set, which are then combined in a log-linear model. To allow an easy integration of new features, the baseline system provides an n-best list of candidate translations which is then reranked using the new features. This framework allows us to incorporate different types of features, including features based on syntactic analyses of the source and target sentences, which we hope will address the grammaticality of the translations, as well as lower-level features. As we work on n-best lists, we can easily use global sentence-level features.

We begin by describing our baseline system and the n-best rescoring framework within which we conducted our experiments. We then present a selection of new features, progressing from word-level features to those based to part-of-speech tags and syntactic chunks, and then to features based on Treebank-based syntactic parses of the source and target sentences.

2 Log-linear Models for Statistical MT

The goal is the translation of a text given in some source language into a target language. We are given a source ('Chinese') sentence $\mathbf{f} = f_1^J = f_1, \ldots, f_j, \ldots, f_J$, which is to be translated into a target ('English') sentence $\mathbf{e} = e_1^I = e_1, \ldots, e_i, \ldots, e_I$ Among all possible target sentences, we will choose the sentence with the highest probability:

$$\hat{e}_1^I = \underset{e_1^I}{\operatorname{argmax}} \{Pr(e_1^I|f_1^J)\} \quad (1)$$

As an alternative to the often used source-channel approach (Brown et al., 1993), we directly model the posterior probability $Pr(e_1^I|f_1^J)$ (Och and Ney, 2002) using a log-linear combination of feature functions. In this framework, we have a set of M feature functions $h_m(e_1^I, f_1^J), m = 1, \ldots, M$. For each feature function, there exists a model parameter $\lambda_m, m = 1, \ldots, M$. The direct translation probability is given by:

$$Pr(e_1^I|f_1^J) = \frac{\exp[\sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J)]}{\sum_{e'^I_1} \exp[\sum_{m=1}^M \lambda_m h_m(e'^I_1, f_1^J)]} \quad (2)$$

We obtain the following decision rule:

$$\hat{e}_1^I = \underset{e_1^I}{\operatorname{argmax}} \left\{ \sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J) \right\} \quad (3)$$

The standard criterion for training such a log-linear model is to maximize the probability of the parallel training corpus consisting of S sentence pairs $\{(\mathbf{f}_s, \mathbf{e}_s) : s = 1, \ldots, S\}$. However, this does not guarantee optimal performance on the metric of translation quality by which our system will ultimately be evaluated. For this reason, we optimize the parameters directly against the BLEU metric on held-out data. This is a more difficult optimization problem, as the search space is no longer convex.

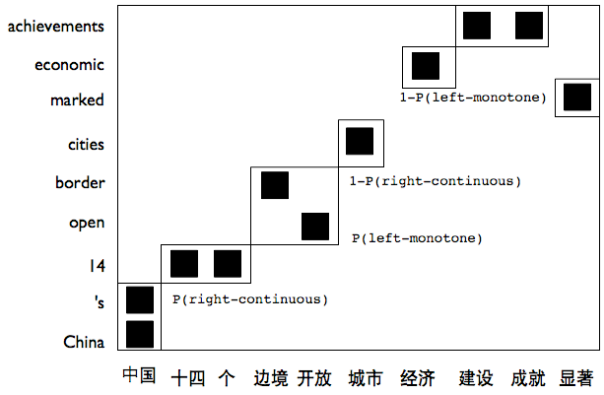


Figure 1: Example segmentation of Chinese sentence and its English translation into alignment templates.

However, certain properties of the BLEU metric can be exploited to speed up search, as described in detail by Och (2003). We use this method of optimizing feature weights throughout this paper.

2.1 Baseline MT System: Alignment Templates

Our baseline MT system is the alignment template system described in detail by Och, Tillmann, and Ney (1999) and Och and Ney (2004). In the following, we give a short description of this baseline model.

The probability model of the alignment template system for translating a sentence can be thought of in distinct stages. First, the source sentence words f_1^J are grouped to phrases $\tilde{f}_1^K$. For each phrase $\tilde{f}$ an alignment template z is chosen and the sequence of chosen alignment templates is reordered (according to π_1^K). Then, every phrase $\tilde{f}$ produces its translation $\tilde{e}$ (using the corresponding alignment template z). Finally, the sequence of phrases $\tilde{e}_1^K$ constitutes the sequence of words e_1^I.

Our baseline system incorporated the following feature functions:

Alignment Template Selection Each alignment template is chosen with probability $p(z|\tilde{f})$, estimated by relative frequency. The corresponding feature function in our log-linear model is the log probability of the product of $p(z|\tilde{f})$ for all used alignment templates used.

Word Selection This feature is based on the lexical translation probabilities $p(e|f)$, estimated using relative frequencies according to the highest-probability word-level alignment for each training sentence. A translation probability conditioned on the source and target position within the alignment template $p(e|f,i,j)$ is interpolated with the position-independent probability $p(e|f)$.

Phrase Alignment This feature favors monotonic alignment at the phrase level. It measures the 'amount of non-monotonicity' by summing over the distance (in the source language) of alignment templates which are consecutive in the target language.

Language Model Features As a language model feature, we use a standard backing off word-based trigram language model (Ney, Generet, and Wessel, 1995). The baseline system actually includes four different language model features trained on four different corpora: the news part of the bilingual training data, a large Xinhua news corpus, a large AFP news corpus, and a set of Chinese news texts downloaded from the web.

Word/Phrase Penalty This word penalty feature counts the length in words of the target sentence. Without this feature, the sentences produced tend to be too short. The phrase penalty feature counts the number of phrases produced, and can allow the model to prefer either short or long phrases.

Phrases from Conventional Lexicon The baseline alignment template system makes use of the Chinese-English lexicon provided by LDC. Each lexicon entry is a potential phrase translation pair in the alignment template system. To score the use of these lexicon entries (which have no normal translation probability), this feature function counts the number of times such a lexicon entry is used.

Additional Features A major advantage of the log-linear modeling approach is that it is easy to add new features. In this paper, we explore a variety of features based on successively deeper syntactic representations of the source and target sentences, and their alignment. For each of the new features discussed below, we added the feature value to the set of baseline features, re-estimated feature weights on development data, and obtained results on test data.

3 Experimental Framework

We worked with the Chinese-English data from the recent evaluations, as both large amounts of sentence-aligned training corpora and multiple gold standard reference translations are available. This is a standard data set, making it possible to compare results with other systems. In addition, working on Chinese allows us to use the existing Chinese syntactic treebank and parsers based on it.

For the baseline MT system, we distinguish the following three different sentence- or chunk-aligned parallel training corpora:

- **training corpus (train)**: This is the basic training corpus used to train the alignment template translation model (word lexicon and phrase lexicon). This corpus consists of about 170M English words. Large parts of this corpus are aligned on a sub-sentence level to avoid the existence of very long sentences which would be filtered out in the training process to allow a manageable word alignment training.
- **development corpus (dev)**: This is the training corpus used in discriminative training of the model-parameters of the log-linear translation model. In

most experiments described in this report this corpus consists of 993 sentences (about 25K words) in both languages.

- **test corpus (test)**: This is the test corpus used to assess the quality of the newly developed feature functions. It consists of 878 sentences (about 25K words).

For development and test data, we have four English (reference) translations for each Chinese sentence.

3.1 Reranking, n-best lists, and oracles

For each sentence in the development, test, and the blind test corpus a set of 16,384 different alternative translations has been produced using the baseline system. For extracting the n-best candidate translations, an A* search is used. These n-best candidate translations are the basis for discriminative training of the model parameters and for re-ranking.

We used n-best reranking rather than implementing new search algorithms. The development of efficient search algorithms for long-range dependencies is very complicated and a research topic in itself. The reranking strategy enabled us to quickly try out a lot of new dependencies, which would not have been be possible if the search algorithm had to be changed for each new dependency.

On the other hand, the use of n-best list rescoring limits the possibility of improvements to what is available in the n-best list. Hence, it is important to analyze the quality of the n-best lists by determining how much of an improvement would be possible given a perfect reranking algorithm. We computed the **oracle translations**, that is, the set of translations from our n-best list that yields the best BLEU score.[1]

We use the following two methods to compute the BLEU score of an oracle translation:

1. optimal oracle (`opt`): We select the oracle sentences which give the highest BLEU score compared **to the set of 4 reference translations**. Then, we compute BLEU score of oracle sentences using **the same set of reference translations**.
2. round-robin oracle (`rr`): We select four different sets of oracle sentences which give the highest BLEU score **compared to each of the 4 references translations**. Then, we compute for each set of oracle sentences a BLEU score **using always those three references to score that have not been chosen to select the oracle**. Then, these 4 3-reference BLEU scores are averaged.

[1]Note that due to the corpus-level holistic nature of the BLEU score it is not trivial to compute the optimal set of oracle translations. We use a greedy search algorithm for the oracle translations that might find only a local optimum. Empirically, we do not observe a dependence on the starting point, hence we believe that this does not pose a significant problem.

Table 1: Oracle BLEU scores for different sizes of the n-best list. The avBLEUr3 scores are computed with respect to three reference translations averaged over the four different choices of holding out one reference.

	avBLEUr3[%]		BLEUr4
n	`rr`	`opt`	`opt`
human	35.8		-
1	28.3	28.3	31.6
4	29.1	30.8	34.5
16	29.9	33.2	37.3
64	30.6	35.6	38.7
256	31.3	37.8	42.8
1024	31.7	40.0	45.3
4096	32.0	41.8	47.3

The first method provides the theoretical upper bound of what BLEU score can be obtained by rescoring a given n-best list. Using this method with a 1000-best list, we obtain oracle translations that outperform the BLEU score of the human translations. The oracle translations achieve 113% against the human BLEU score on the test data (Table 1), while the first best translations obtain 79.2% against the human BLEU score. The second method uses a different references for selection and scoring. Here, using an 1000-best list, we obtain oracle translations with a relative human BLEU score of 88.5%.

Based on the results of the oracle experiment, and in order to make rescoring computationally feasible for features requiring significant computation for each hypothesis, we used the top 1000 translation candidates for our experiments. The baseline system's BLEU score is 31.6% on the test set (equivalent to the 1-best oracle in Table 1). This is the benchmark against which the contributions of the additional features described in the remainder of this paper are to be judged.

3.2 Preprocessing

As a precursor to developing the various syntactic features described in this report, the syntactic representations on which they are based needed to be computed. This involved part-of-speech tagging, chunking, and parsing both the Chinese and English side of our training, development, and test sets.

Applying the part-of-speech tagger to the often ungrammatical MT output from our n-best lists sometimes led to unexpected results. Often the tagger tries to "fix up" ungrammatical sentences, for example by looking for a verb when none is present:

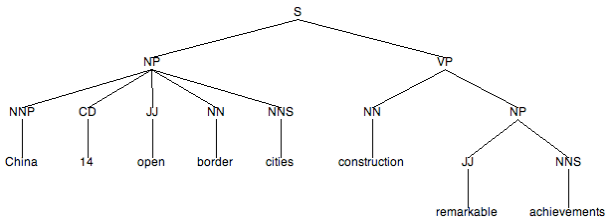
China_NNP 14_CD open_JJ border_NN
cities_NNS **achievements_VBZ** remarkable_JJ

Here, although *achievements* has never been seen as a verb in the tagger's training data, the prior for a verb in this position is high enough to cause a present tense

verb tag to be produced. In addition to the inaccuracies of the MT system, the difference in genre from the tagger's training text can cause problems. For example, while our MT data include news article headlines with no verb, headlines are not included in the Wall Street Journal text on which the tagger is trained. Similarly, the tagger is trained on full sentences with normalized punctuation, leading it to expect punctuation at the end of every sentence, and produce a punctuation tag even when the evidence does not support it:

> China_NNP 's_POS economic_JJ
> development_NN and_CC opening_VBG
> up_RP 14_CD border_NN cities_NNS
> remarkable_JJ **achievements_.**

The same issues affect the parser. For example the parser can create verb phrases where none exist, as in the following example in which the tagger correctly did not identify a verb in the sentence:



These effects have serious implications for designing syntactic feature functions. Features such "is there a verb phrase" may not do what you expect. One solution would be features that involve the probability of a parse subtree or tag sequence, allowing us to ask "how good a verb phrase is it?" Another solution is more detailed features examining more of the structure, such as "is there a verb phrase *with a verb*?"

4 Word-Level Feature Functions

These features, directly based on the source and target strings of words, are intended to address such problems as translation choice, missing content words, and incorrect punctuation.

4.1 Model 1 Score

We used IBM Model 1 (Brown et al., 1993) as one of the feature functions. Since Model 1 is a bag-of-word translation model and it gives the sum of all possible alignment probabilities, a lexical co-occurrence effect, or *triggering effect*, is expected. This captures a sort of topic or semantic coherence in translations.

As defined by Brown et al. (1993), Model 1 gives a probability of any given translation pair, which is

$$p(\mathbf{f}|\mathbf{e};\mathrm{M1}) = \frac{\epsilon}{(l+1)^m}\prod_{j=1}^{m}\sum_{i=0}^{l} t(f_j|e_i).$$

We used GIZA++ to train the model. The training data is a subset (30 million words on the English side) of the entire corpus that was used to train the baseline MT system. For a missing translation word pair or unknown words, where $t(f_j|e_i) = 0$ according to the model, a constant $t(f_j|e_i) = 10^{-40}$ was used as a smoothing value.

The average %BLEU score (average of the best four among different 20 search initial points) is 32.5. We also tried $p(\mathbf{e}|\mathbf{f};\mathrm{M1})$ as feature function, but did not obtain improvements which might be due to an overlap with the word selection feature in the baseline system.

The Model 1 score is one of the best performing features. It seems to 'fix' the tendency of our baseline system to delete content words and it improves word selection coherence by the triggering effect. It is also possible that the triggering effect might work on selecting a proper verb-noun combination, or a verb-preposition combination.

4.2 Lexical Re-ordering of Alignment Templates

As shown in Figure 1 the alignment templates (ATs) used in the baseline system can appear in various configurations which we will call *left/right-monotone* and *left/right-continuous*. We built 2 out of these 4 models to distinguish two types of lexicalized re-ordering of these ATs:

The **left-monotone model** computes the total probability of all ATs being left monotone: where the lower left corner of the AT touches the upper right corner of the previous AT. Note that the first word in the current AT may or may not immediately follow the last word in the previous AT. The total probability is the product over all alignment templates i, either $P(AT_i$ *is left-monotone*$)$ or $1 - P(AT_i$ *is left-monotone*$)$.

The **right-continuous model** computes the total probability of all ATs being right continuous: where the lower left corner of the AT touches the upper right corner of the previous AT *and* the first word in the current AT immediately follows the last word in the previous AT. The total probability is the product over all alignment templates i, either $P(AT_i$ *is right-continuous*$)$ or $1 - P(AT_i$ *is right-continuous*$)$.

In both models, the probabilities P have been estimated from the full training data (**train**).

5 Shallow Syntactic Feature Functions

By shallow syntax, we mean the output of the part-of-speech tagger and chunkers. We hope that such features can combine the strengths of tag- and chunk-based translation systems (Schafer and Yarowsky, 2003) with our baseline system.

5.1 Projected POS Language Model

This feature uses Chinese POS tag sequences as surrogates for Chinese words to model movement. Chinese words are too sparse to model movement, but an attempt

to model movement using Chinese POS may be more successful. We hope that this feature will compensate for a weak model of word movement in the baseline system.

Chinese POS sequences are projected to English using the word alignment. Relative positions are indicated for each Chinese tag. The feature function was also tried without the relative positions:

CD_+0_M_+1	NN_+3	NN_-1	NN_+2_NN_+3
14 $(measure)$	open	border	cities

The table shows an example tagging of an English hypothesis showing how it was generated from the Chinese sentence. The feature function is the log probability output by a trigram language model over this sequence. This is similar to the HMM Alignment model (Vogel, Ney, and Tillmann, 1996) but in this case movement is calculated on the basis of parts of speech.

The Projected POS feature function was one of the strongest performing shallow syntactic feature functions, with a %BLEU score of 31.8. This feature function can be thought of as a trade-off between purely word-based models, and full generative models based upon shallow syntax.

6 Tree-Based Feature Functions

Syntax-based MT has shown promise in the work of, among others, Wu and Wong (1998) and Alshawi, Bangalore, and Douglas (2000). We hope that adding features based on Treebank-based syntactic analyses of the source and target sentences will address grammatical errors in the output of the baseline system.

6.1 Parse Tree Probability

The most straightforward way to integrate a statistical parser in the system would be the use of the (log of the) parser probability as a feature function. Unfortunately, this feature function did not help to obtain better results (it actually seems to significantly hurt performance).

To analyze the reason for this, we performed an experiment to test if the used statistical parser assigns a higher probability to presumably grammatical sentences. The following table shows the average log probability assigned by the Collins parser to the 1-best (produced), oracle and the reference translations:

Hypothesis	1-best	Oracle	Reference
log(parseProb)	-147.2	-148.5	-154.9

We observe that the average parser log-probability of the 1-best translation is higher than the average parse log probability of the oracle or the reference translations. Hence, it turns out that the parser is actually assigning higher probabilities to the ungrammatical MT output than to the presumably grammatical human translations. One reason for that is that the MT output uses fewer unseen words and typically more frequent words which lead to a higher language model probability. We also performed experiments to balance this effect by dividing the parser probability by the word unigram probability and using this 'normalized parser probability' as a feature function, but also this did not yield improvements.

6.2 Tree-to-String Alignment

A *tree-to-string* model is one of several syntax-based translation models used. The model is a conditional probability $p(\mathbf{f}|T(\mathbf{e}))$. Here, we used a model defined by Yamada and Knight (2001) and Yamada and Knight (2002).

Internally, the model performs three types of operations on each node of a parse tree. First, it *reorders* the child nodes, such as changing $\mathsf{VP} \rightarrow \mathsf{VB\ NP\ PP}$ into $\mathsf{VP} \rightarrow \mathsf{NP\ PP\ VB}$. Second, it *inserts* an optional word at each node. Third, it *translates* the leaf English words into Chinese words. These operations are stochastic and their probabilities are assumed to depend only on the node, and are independent of other operations on the node, or other nodes. The probability of each operation is automatically obtained by a training algorithm, using about 780,000 English parse tree-Chinese sentence pairs. The probability of these operations $\theta(e^k_{i,j})$ is assumed to depend on the edge of the tree being modified, $e^k_{i,j}$, but independent of everything else, giving the following equation,

$$p(\mathbf{f}|T(\mathbf{e})) = \sum_{\Theta} \prod_{\theta(e^k_{i,j})} p(\theta(e^k_{i,j})|e^k_{i,j}) \qquad (4)$$

where Θ varies over the possible alignments between the $\mathbf{f}$ and $\mathbf{e}$ and $\theta(e^k_{i,j})$ is the particular operations (in Θ) for the edge $e^k_{i,j}$.

The model is further extended to incorporate phrasal translations performed at each node of the input parse tree (Yamada and Knight, 2002). An English phrase covered by a node can be directly translated into a Chinese phrase without regular reorderings, insertions, and leaf-word translations.

The model was trained using about 780,000 English parse tree-Chinese sentence pairs. There are about 3 million words on the English side, and they were parsed by Collins' parser.

Since the model is computationally expensive, we added some limitations on the model operations. As the base MT system does not produce a translation with a big word jump, we restrict the model not to reorder child nodes when the node covers more than seven words. For a node that has more than four children, the reordering probability is set to be uniform. We also introduced pruning, which discards partial (subtree-substring) alignments if the probability is lower than a threshold.

The model gives a sum of all possible alignment probabilities for a pair of a Chinese sentence and an English parse tree. We also calculate the probability of the best alignment according to the model. Thus, we have the fol-

lowing two feature functions:

$$h_{\text{TreeToStringSum}}(\mathbf{e}, \mathbf{f}) = \log(\sum_{\Theta} \prod_{\theta(e^k_{i,j})} p(\theta(e^k_{i,j})|e^k_{i,j}))$$

$$h_{\text{TreeToStringViterbi}}(\mathbf{e}, \mathbf{f}) = \log(\max_{\Theta} \prod_{\theta(e^k_{i,j})} p(\theta(e^k_{i,j})|e^k_{i,j}))$$

As the model is computationally expensive, we sorted the n-best list by the sentence length, and processed them from the shorter ones to the longer ones. We used 10 CPUs for about five days, and 273/997 development sentences and 237/878 test sentences were processed.

The average %BLEU score (average of the best four among different 20 search initial points) was 31.7 for both $h_{\text{TreeToStringSum}}$ and $h_{\text{TreeToStringViterbi}}$. Among the processed development sentences, the model preferred the oracle sentences over the produced sentence in 61% of the cases.

The biggest problem of this model is that it is computationally very expensive. It processed less than 30% of the n-best lists in long CPU hours. In addition, we processed short sentences only. For long sentences, it is not practical to use this model as it is.

6.3 Tree-to-Tree Alignment

A *tree-to-tree* translation model makes use of syntactic tree for both the source and target language. As in the tree-to-string model, a set of operations apply, each with some probability, to transform one tree into another. However, when training the model, trees for both the source and target languages are provided, in our case from the Chinese and English parsers.

We began with the tree-to-tree alignment model presented by Gildea (2003). The model was extended to handle dependency trees, and to make use of the word-level alignments produced by the baseline MT system. The probability assigned by the tree-to-tree alignment model, given the word-level alignment with which the candidate translation was generated, was used as a feature in our rescoring system.

We trained the parameters of the tree transformation operations on 42,000 sentence pairs of parallel Chinese-English data from the Foreign Broadcast Information Service (FBIS) corpus. The lexical translation probabilities P_t were trained using IBM Model 1 on the 30 million word training corpus. This was done to overcome the sparseness of the lexical translation probabilities estimated while training the tree-to-tree model, which was not able to make use of as much training data.

As a test of the tree-to-tree model's discrimination, we performed an oracle experiment, comparing the model scores on the first sentence in the n-best list with candidate giving highest BLEU score. On the 1000-best list for the 993-sentence development set, restricting ourselves to sentences with no more than 60 words and a branching factor of no more than five in either the Chinese or English tree, we achieved results for 480, or 48% of the 993 sentences. Of these 480, the model preferred the produced over the oracle 52% of the time, indicating that it does not in fact seem likely to significantly improve BLEU scores when used for reranking. Using the probability of the source Chinese dependency parse aligning with the n-best hypothesis dependency parse as a feature function, making use of the word-level alignments, yields a 31.6 %BLEU score — identical to our baseline.

6.4 Markov Assumption for Tree Alignments

The tree-based feature functions described so far have the following limitations: full parse tree models are expensive to compute for long sentences and for trees with flat constituents and there is limited reordering observed in the n-best lists that form the basis of our experiments. In addition to this, higher levels of parse tree are rarely observed to be reordered between source and target parse trees.

In this section we attack these problems using a simple Markov model for tree-based alignments. It guarantees tractability: compared to a coverage of approximately 30% of the n-best list by the unconstrained tree-based models, using the Markov model approach provides 98% coverage of the n-best list. In addition, this approach is robust to inaccurate parse trees.

The algorithm works as follows: we start with word alignments and two parameters: n for maximum number of words in tree fragment and k for maximum height of tree fragment. We proceed from left to right in the Chinese sentence and incrementally grow a pair of subtrees, one subtree in Chinese and the other in English, such that each word in the Chinese subtree is aligned to a word in the English subtree. We grow this pair of subtrees until we can no longer grow either subtree without violating the two parameter values n and k. Note that these aligned subtree pairs have properties similar to alignment templates. They can rearrange in complex ways between source and target. Figure 2 shows how subtree-pairs for parameters $n = 3$ and $k = 3$ can be drawn for this sentence pair. In our experiments, we use substantially bigger tree fragments with parameters set to $n = 8$ and $k = 9$.

Once these subtree-pairs have been obtained, we can easily assert a Markov assumption for the tree-to-tree and tree-to-string translation models that exploits these pairings. Let consider a sentence pair in which we have discovered n subtree-pairs which we can call Frag_0, ..., Frag_n. We can then compute a feature function for the sentence pair using the tree-to-string translation model as follows:

$$h_{\mathit{MarkovTreeToString}} = \mathit{logP}_{\mathit{tree\text{-}to\text{-}string}}(\mathit{Frag}_0) + \ldots + \mathit{logP}_{\mathit{tree\text{-}to\text{-}string}}(\mathit{Frag}_n)$$

Using this Markov assumption on tree alignments with

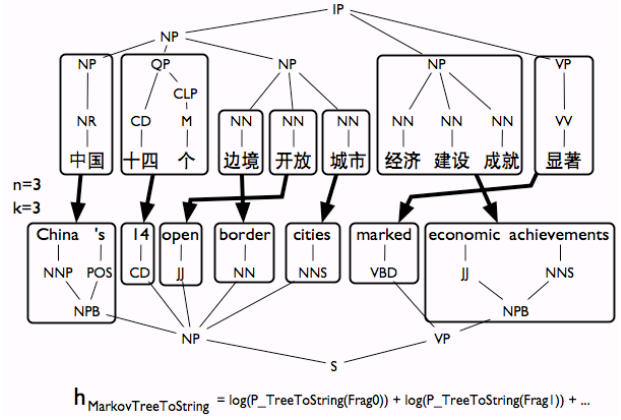


Figure 2: Markov assumption on tree alignments.

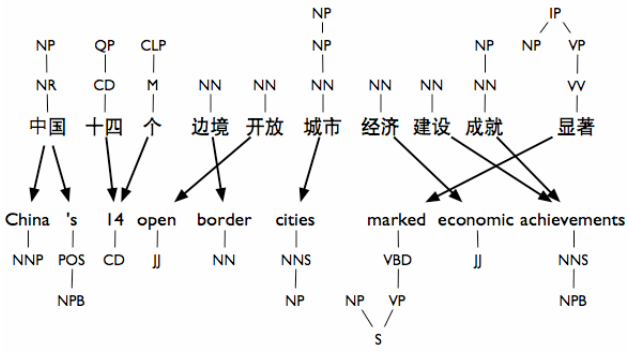


Figure 3: Word alignments with TAG elementary trees.

the Tree to String model described in Section 6.2 we obtain a coverage improvement to 98% coverage from the original 30%. The accuracy of the tree to string model also improved with a %BLEU score of 32.0 which is the best performing single syntactic feature.

6.5 Using TAG elementary trees for scoring word alignments

In this section, we consider another method for carving up the full parse tree. However, in this method, instead of subtree-pairs we consider a decomposition of parse trees that provides each word with a fragment of the original parse tree as shown in Figure 3. The formalism of Tree-Adjoining Grammar (TAG) provides the definition what each tree fragment should be and in addition how to decompose the original parse trees to provide the fragments. Each fragment is a TAG elementary tree and the composition of these TAG elementary trees in a TAG derivation tree provides the decomposition of the parse trees. The decomposition into TAG elementary trees is done by augmenting the parse tree for source and target sentence with head-word and argument (or complement) information using heuristics that are common to most contemporary statistical parsers and easily available for both English and Chinese. Note that we do not use the word alignment information for the decomposition into TAG elementary trees.

Once we have a TAG elementary tree per word, we can create several models that score word alignments by exploiting the alignments between TAG elementary trees between source and target. Let t_{f_i} and t_{e_i} be the TAG elementary trees associated with the aligned words f_i and e_i respectively. We experimented with two models over alignments: unigram model over alignments: $\prod_i P(f_i, t_{f_i}, e_i, t_{e_i})$ and conditional model: $\prod_i P(e_i, t_{e_i} \mid f_i, t_{f_i}) \times P(f_{i+1}, t_{f_{i+1}} \mid f_i, t_{f_i})$

We trained both of these models using the SRI Language Modeling Toolkit using 60K aligned parse trees. We extracted 1300 TAG elementary trees each for Chinese and for English. The unigram model gets a %BLEU score of 31.7 and the conditional model gets a %BLEU score of 31.9.

	%BLEU
Baseline	31.6
IBM Model 1 $p(\mathbf{f}\|\mathbf{e})$	32.5
Tree-to-String Markov fragments	32.0
Right-continuous alignment template	32.0
TAG conditional bigrams	31.9
Left-monotone alignment template	31.9
Projected POS LM	31.8
Tree-to-String	31.7
TAG unigram	31.7
Tree-to-Tree	31.6
combination	32.9

Table 2: Results for the baseline features, each new feature added to the baseline features on its own, and a combination of new features.

7 Conclusions

The use of discriminative reranking of an n-best list produced with a state-of-the-art statistical MT system allowed us to rapidly evaluate the benefits of off-the-shelf parsers, chunkers, and POS taggers for improving syntactic well-formedness of the MT output. Results are summarized in Table 2; the best single new feature improved the %BLEU score from 31.6 to 32.5. The 95% confidence intervals computed with the bootstrap resampling method are about 0.8%. In addition to experiments with single features we also integrated multiple features using a greedy approach where we integrated at each step the feature that most improves the BLEU score. This feature integration produced a statistically significant improvement of absolute 1.3% to 32.9 %BLEU score.

Our single best feature, and in fact the only single feature to produce a truly significant improvement, was the IBM Model 1 score. We attribute its success that it addresses the weakness of the baseline system to omit con-

tent words and that it improves word selection by employing a triggering effect. We hypothesize that this allows for better use of context in, for example, choosing among senses of the source language word.

A major goal of this work was to find out if we can exploit annotated data such as treebanks for Chinese and English and make use of state-of-the-art deep or shallow parsers to improve MT quality. Unfortunately, none of the implemented syntactic features achieved a statistically significant improvement in the BLEU score. Potential reasons for this might be:

- As described in Section 3.2, the use of off-the-shelf taggers and parsers has various problems due to various mismatches between the parser training data and our application domain. This might explain that the use of the parser probability as feature function was not successful. A potential improvement might be to adapt the parser by retraining it on the full training data that has been used by the baseline system.
- The use of a 1000-best list limits the potential improvements. It is possible that more improvements could be obtained using a larger n-best list or a word graph representation of the candidates.
- The BLEU score is possibly not sufficiently sensitive to the grammaticality of MT output. This could not only make it difficult to see an improvement in the system's output, but also potentially mislead the BLEU-based optimization of the feature weights. A significantly larger corpus for discriminative training and for evaluation would yield much smaller confidence intervals.
- Our discriminative training technique, which directly optimizes the BLEU score on a development corpus, seems to have overfitting problems with large number of features. One could use a larger development corpus for discriminative training or investigate alternative discriminative training criteria.
- The amount of annotated data that has been used to train the taggers and parsers is two orders of magnitude smaller than the parallel training data that has been used to train the baseline system (or the word-based features). Possibly, a comparable amount of annotated data (e.g. a treebank with 100 million words) is needed to obtain significant improvements.

This is the first large scale integration of syntactic analysis operating on many different levels with a state-of-the-art phrase-based MT system. The methodology of using a log-linear feature combination approach, discriminative reranking of n-best lists computed with a state-of-the-art baseline system allowed members of a large team to simultaneously experiment with hundreds of syntactic feature functions on a common platform.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 0121285.

References

Alshawi, Hiyan, Srinivas Bangalore, and Shona Douglas. 2000. Learning dependency translation models as collections of finite state head transducers. *Computational Linguistics*, 26(1):45–60.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Gildea, Daniel. 2003. Loosely tree-based alignment for machine translation. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan.

Ney, Hermann, M. Generet, and Frank Wessel. 1995. Extensions of absolute discounting for language modeling. In *Proc. of the Fourth European Conf. on Speech Communication and Technology*, Madrid, Spain.

Och, Franz Josef. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan.

Och, Franz Josef and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA.

Och, Franz Josef and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*. Accepted for Publication.

Och, Franz Josef, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*, College Park, MD.

Schafer, Charles and David Yarowsky. 2003. Statistical machine translation using coercive two-level syntactic transduction. In *Proc. of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, PA.

Vogel, Stephan, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING '96: The 16th Int. Conf. on Computational Linguistics*, Copenhagen, Denmark.

Wu, Dekai and H. Wong. 1998. Machine translation with a stochastic grammatical channel. In *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th Int. Conf. on Computational Linguistics*, Montreal, Canada.

Yamada, Kenji and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proc. of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, Toulouse, France.

Yamada, Kenji and Kevin Knight. 2002. A decoder for syntax-based MT. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA.