Source-Side Classifier Preordering for Machine Translation

Uri Lerner
Google Inc.
Mountain View, CA, USA
uri@google.com

Slav Petrov
Google Inc.
New York, NY, USA
slav@google.com

Abstract

We present a simple and novel classifier-based preordering approach. Unlike existing preordering models, we train feature-rich discriminative classifiers that directly predict the target-side word order. Our approach combines the strengths of lexical reordering and syntactic preordering models by performing long-distance reorderings using the structure of the parse tree, while utilizing a discriminative model with a rich set of features, including lexical features. We present extensive experiments on 22 language pairs, including preordering into English from 7 other languages. We obtain improvements of up to 1.4 BLEU on language pairs in the WMT 2010 shared task. For languages from different families the improvements often exceed 2 BLEU. Many of these gains are also significant in human evaluations.

1 Introduction

Generating the appropriate word order for the target language has been one of the fundamental problems in machine translation since the ground setting work of Brown et al. (1990). Lexical reordering approaches (Tillmann, 2004; Zens and Ney, 2006) add a reordering component to standard phrase-based translation systems (Och and Ney, 2004). Because the reordering model is trained discriminatively, it can use a rich set of lexical features. However, it only has access to the local context which often times is insufficient to make the long-distance reordering decisions that are necessary for language pairs with significantly different word order.

Preordering (sometimes called pre-reordering or simply reordering) approaches (Xia and McCord, 2004; Collins et al., 2005) preprocess the input in such a way that the words on the source side appear closer to their final positions on the target side. Because preordering is performed prior to word alignment, it can improve the alignment process and can then be combined with any subsequent translation model. Most preordering models use a source-side syntactic parser and perform a series of tree transformations. Approaches that do not use a parser exist as well and typically induce a hierarchical representation that also allows them to perform long-distance changes (Tromble and Eisner, 2009; DeNero and Uszkoreit, 2011; Neubig et al., 2012).

Models that use a source-side parser differ on two main dimensions: the way tree transformations are expressed, and whether they are built manually or learned from data. One common type of tree transformation are rewrite rules. These typically involve some condition under which the transformation can be applied (e.g., a noun and an adjective found in the same clause) and the transformation itself (e.g., move the adjective after the noun). These rules can be designed manually (Collins et al., 2005; Wang et al., 2007) or learned from data (Xia and McCord, 2004; Habash, 2007; Genzel, 2010; Wu et al., 2011).

Another type of tree transformations uses ranking functions to implement precedence-based reordering. Here, a function assigns a numerical value to every word in a clause, intended to express the precedence of the word in the target language. The reordering operation is then to sort the words according to their assigned values. The ranking function

Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 513–523,
Seattle, Washington, USA, 18-21 October 2013. ©2013 Association for Computational Linguistics

can be designed manually (Xu et al., 2009) or trained from data (Yang et al., 2012). This approach is particularly effective for Subject-Object-Verb (SOV) languages.

In this work we present a simple classifier-based preordering model. Our model operates over dependency parse trees and is therefore able to perform long-distance reordering decisions, as is typical for preordering models. But instead of deterministic rules or ranking functions, we use discriminative classifiers to directly predict the final word order, using rich (bi-)lexical and syntactic features.

We present two models. The first model uses a classifier to directly predict the permutation order in which a family of words (a head word and all its children) will appear on the target side. This approach is similar in spirit to the work of Li et al. (2007), except that they use constituency parse trees and consider only nodes with 2 or 3 children. We instead work with dependency trees and consider much larger head-children sets. Our second model is designed to decompose the exponential search space of all possible permutations. The prediction task is broken into two separate steps. In the first step, for each child word a binary classifier decides whether it appears before or after its parent in the target language. In the second step, we predict the best order of the words on each side of the parent. We show that the second approach is never worse than the first one and sometimes significantly better.

We present experiments on 22 language pairs from different language families using our preordering approach in a phrase-based system (Och and Ney, 2004), as well as a forest-to-string system (Zhang et al., 2011). In a first set of experiments, we use the WMT 2010 shared task data (Callison-Burch et al., 2010) and show significant improvements of up to 1.4 BLEU (Papineni et al., 2002) on three out of eight language pairs. In a second set of experiments, we use automatically mined parallel data from the web and build translation systems for languages from various language families. We obtain especially big improvements in translation quality (2-7 BLEU) when the language pairs have divergent word order (for example English to Indonesian, Japanese, Korean or Malay). In our experiments on English to and from Hungarian, Dutch, and Portuguese translation, we find that we can obtain consistent improvements in both translation directions. To additionally verify our improvements we use human raters, who confirm the significance of the BLEU score improvements.

Finally, we compare training the preordering classifiers on small amounts of manually aligned data to training on large quantities of automatically aligned data for English to Arabic, Hebrew, and Japanese. When evaluated on a pure reordering task, the models trained on manually aligned data perform slightly better, but similar BLEU scores are obtained in both scenarios on an end-to-end translation task.

2 Classifier Reordering

Our goal is to learn a model that can transform the word order of an input sentence to an order that is natural in the target language. For example, when translating the English sentence:

The black cat climbed to the tree top.

to Spanish, we would like to reorder it as:

The cat black climbed to the top tree.

When translating to Japanese, we would like to get:

The black cat the tree top to climbed.

Such a model can then be used in combination with any translation model.

In our approach we first part-of-speech (POS) tag and parse the input sentence, producing the POS tags and head-modifier dependencies shown in Figure 1. Reordering is then done by traversing the dependency tree starting at the root. For each head word we determine the order of the head and its children (independently of other decisions) and continue the traversal recursively in that order. In the example, we first need to decide on the order of the head *"climbed"* and the children *"cat"*, *"to"*, and *"."*.

2.1 Classification Model & Features

The reordering decisions are made by multi-class classifiers where class labels correspond to permutation sequences. We train a separate classifier for each number of possible children. Crucially, we do not learn explicit tree transformations rules, but let the classifiers learn to trade off between a rich set of overlapping features.

Obviously, it is possible to use any classification model and learning algorithm. We use maximum entropy classifiers with l_1/l_∞ regularization trained with the GradBoost algorithm (Duchi and Singer, 2009). We chose this setup since it naturally supports multi-class prediction and can therefore be used to select one out of many possible permutations. Additionally, the learning algorithm produces a sparse set of features. In our experiments the final models have typically only a few 100K non-zero feature weights per language pair. Given this relatively small number of features, it is possible to manually inspect the feature weights and gain insights into the behavior of the model. We show an example analysis in Section 5.

Our features encode information about the context in which a word occurs in the sentence. We model context as "informative" words:

- The head itself.
- The children. We indicate whether each child is before, immediately before, immediately after, or after the head.
- For every child, if there is a gap between it and the head, then the first and last word of that gap.
- For every pair of consecutive children, if there is a gap between them, then the first and last word of that gap.
- The head's immediate sibling to the left/right or an indication that none exists.

When extracting the features, every word can be represented by its word identity, its fine-grained POS tag from the treebank, and a coarse-grained POS category, similar to the universal categories described in Petrov et al. (2012). We also include pairs of these features, resulting in potentially bilexical features.

2.2 Training Data

The training data for the classifiers is generated from the word aligned parallel text. Since parallel data is plentiful, we can afford to be selective. We first construct the intersection of high-confidence source-to-target and target-to-source alignments. For every family in the source dependency tree we generate a training instance if and only if the intersection defines a full order on the source words:

- Every source word must be aligned to at least one target word.
- No two source words can be aligned to the same target word.
- If a source word is aligned to multiple target words, then no target word in this range can be aligned to a different source word.

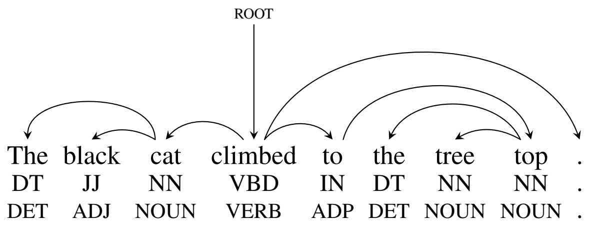
ROOT
The black cat climbed to the tree top .
DT JJ NN VBD IN DT NN NN .
DET ADJ NOUN VERB ADP DET NOUN NOUN .

Figure 1: A sentence, its dependency parse and its fine-grained and coarse-grained POS tags.

While this might sound restrictive, we can usually generate at least some training instances from every sentence and discard the remaining families in the tree. In particular, we do not need to extract training instances for all words in a given sentence since the reordering decisions are made independently for every head word.

A potential concern might be that our method for selecting training data can exclude all instances of certain words. Consider the English phrase *"the boy"*. For languages without articles (e.g. Russian or Japanese) the determiner *"the"* may either not be aligned to any word or get aligned to the foreign word for *"boy"*. In both cases the family will be discarded according to either the first or the second condition above. The concern is therefore that we would have no training data with the English word *"the"*. In practice, however, this does not seem to be a problem. First, there are instances where the English word *"the"* gets aligned to something (perhaps a preposition), and second, since the word *"the"* is omitted in the target language its location in the reordered sentence is not very important.

Naturally we learn better classifier models from better alignments. The other direction is also true – if we run preordering on the source side then the alignment task becomes easier and tends to produce better results. Therefore it can be useful to iterate between generating the alignment and learning a preordering model. Empirically, the gains from this bootstrapping approach are not dramatic and are realized after just one iteration, i.e., create the alignment, train a preordering model, use the preordering model to learn a new alignment, and then train the final preordering model.

2.3 1-Step Classifier

As a first approach we use a single classifier to directly predict the correct permutation of a given family. Consider the family headed by *"climbed"* in Figure 1. There are three children and the original order of the words is *"cat"*, *"climbed"*, *"to"*, and *"."*. A possible outcome of the classifier can be the permutation 0-2-1-3, representing the order *"cat"*, *"to"*, *"climbed"*, and *"."*.

The number of permutations for the head and n children is of course $(n + 1)!$, which becomes large very quickly and causes some problems. In practice we therefore limit ourselves to the K most common permutations. Unfortunately, this means that whenever there are many children, the correct permutation order might not be available as an option. Even when the correct permutation is available, classification accuracy typically deteriorates as the number of possible classes increases.

An additional subtle issue is that the 1-step classifier cannot share useful information across different numbers of children. For example, in Spanish adjectives usually appear after the noun, but sometimes they appear before the noun. The decision depends on the adjective itself and sometimes the head noun, but does not depend on other children. Ideally, if for some adjective we have enough examples with 1 or 2 children we would like to make the same decision for a larger number of children, but these classifiers may not have enough relevant examples.

2.4 2-Step Classifier

Our 2-step approach addresses the exponential blowup of the number of children by decomposing the prediction into two steps:

1. For every child, decide whether it should appear before or after the head.
2. Determine the order of the children that appear before the head and the order of the children after the head.

The two steps make the reordering of the modifiers before and after the head independent of each other, which is reminiscent of the lexicalized parse tree generation approach of Collins (1997). In the running example, for the head *"climbed"* we might first make the following three binary decisions: the word *"cat"* should appear before the head and the words *"to"* and *"."* should appear after the head. In the second step there is only one word before the head so there is nothing to do. There are two words after the head, so we use another classifier to determine their order. The first step is implemented using a binary classifier, called the *pivot* classifier (since the head functions like the pivot in quicksort). The second step classifiers directly predict the correct permutation of the children before / after the head.

To illustrate the effectiveness of the 2-step approach, consider a head word with 4 children. The 1-step approach must predict 1 of $5! = 120$ outcomes. In the 2-step approach, in the worst case the second step must predict 1 of $4! = 24$ outcomes (if all the children are on one side of the head); if we are lucky and the children split evenly, then we only need two binary decisions in the second step (for the two pairs before and after the head). If we define hard cases as cases involving 5 or more words, 5.54% of the non-leaves are hard cases with the 1-step approach, but only 1.07% are hard cases with the 2-step approach.

3 Experimental Setup

To provide a through evaluation of our approach, we conduct experiments on two sets of data and with two translation systems. The first translation system is a phrase-based system (Och and Ney, 2004). In addition to the regular distance distortion model, we incorporate a maximum entropy based lexicalized phrase reordering model (Zens and Ney, 2006). Our second system is a forest-to-string system (Zhang et al., 2011). The forest-to-string system uses a one-best parse tree but factorizes it into a packed forest of binary elementary trees – hence the name forest-to-string rather than tree-to-string.

The systems are configured and tuned for each language pair to produce the best results. We then add our 1-step and 2-step preordering classifiers as preprocessing steps at training and test time. We train the reordering classifiers on up to 15M training instances. We train separate classifiers for every number of involved words, and restrict each one to the $K = 20$ most frequent outcomes.

In our implementation, in the 1-step approach we did not do any reordering for nodes with 7 or more children. In the 2-step approach we did not reorder the children on either side of the head if there were 7 or more of them. Even though there was no technical reason that prevented us from raising the thresholds, there was no good reason to do so. There were very few cases where children were not reordered because of these thresholds, many of them corresponded to bad parses, and they had very little impact on the final scores. Thus, for the 1-step approach we had 6 classifiers: 1 binary classifier for a head and a single child and 5 multi-class classifiers for 3–7 words. For the 2-step approach we had 11 classifiers: 1 pivot classifier, 5 classifiers for words before the head, and 5 for words after the head.

For a direct comparison to a strong preordering system, we compare to the system of Genzel (2010), which learns a set of unlexicalized reordering rules from automatically aligned data by minimizing the number of crossing alignments. We used a sliding window of size 3 and tried all three of their variants. There were about 40-50 rules per language pair. While conceptually possible, it is not practical to learn more rules (including lexicalized rules) with this system, because of the computational complexity of the learning algorithm and the incremental nature in which the rules are learned and applied.

3.1 WMT Setup

In our first set of experiments, we use the data provided for the WMT 2010 shared task (Callison-Burch et al., 2010). We build systems for all language pairs: English to and from Czech, French, German, and Spanish. Since this is a publicly available dataset, it is easy to compare our results to other submissions to the shared task.

During word alignment, we filter out sentences exceeding 60 words in the parallel texts and perform 6 iterations of IBM Model-1 training (Brown et al., 1993), followed by 6 iterations of HMM training (Vogel et al., 1996). We do not use Model-4 because it is slow and did not add much value to our systems in a pilot study. Standard phrase extraction heuristics (Koehn et al., 2003) are applied to extract phrase pairs with a length limit of 6 from alignments symmetrized with the "union" heuristic. Maximum jump width is set to 8. Rule extraction for the forest-to-string system is limited to 16 rules per tree node. There are no length-based reordering constraints in the forest-to-string system. We train two 5-gram language models with Kneser-Ney smoothing for each of the target languages. One is trained on the target side of the parallel text, the other on a news corpus provided by the shared task. We tune the feature weights for every configuration with 10 rounds of hypergraph-based Minimum Error Rate Training (MERT) (Kumar et al., 2009).

3.2 Additional Languages

In our second set of experiments, we explore the impact of classifier preordering for a number of languages with different word orders. Some of the languages included in our study are verb-subject-object (VSO) languages (Arabic, Irish, Welsh), subject-object-verb (SOV) languages (Japanese, Korean), and fairly free word order languages (Dutch, Hungarian). Where a parser is available, we also conduct experiments on translating into English.

Since there are no standard training sets for many of these language pairs, we use parallel data automatically mined from the web. The amount of parallel text for each language pair is between 120M and 160M words. For evaluation, we use a set of 9K English sentences collected from the web and translated by humans into each of the target languages. Each sentence has one reference translation. We use 5K sentences for evaluation and the rest for tuning.

The systems and training configurations are similar to the WMT setup. The word alignment step includes 3 iterations of IBM Model-1 training and 2 iterations of HMM training. Lexical reordering is included where it helps, but typically makes only a small difference. We again use a 5-gram language model trained on a large amount of monolingual text. Overall, we use between 20 and 30 features, whose weights are optimized using hypergraph-based MERT. All experiments for a given language pair use the same set of MERT weights. This potentially underestimates the improvements that can be obtained, but also eliminates MERT as a possible source of improvement, allowing us to trace back improvements in translation quality directly to changes in preordering of the input data.

3.3 Evaluation

We use case-sensitive BLEU (Papineni et al., 2002) to assess translation quality. For Japanese and Korean we use character-level BLEU. We use bootstrap resampling to compute confidence intervals.

Additionally, we also conduct a side-by-side human evaluation on 750 sentences for each language pair (sampled from the same sentences used for computing BLEU). For each sentence, we ask bilingual annotators to compare the translations from two different systems and say whether one is better, leading to three possible scores of -1, 0, and +1. We focus on this relative comparison since absolute scores are difficult to calibrate across languages and raters.

3.4 Syntactic Parsers

Table 1 shows our treebank sources and parsing accuracies. For English, we use the updated WSJ with OntoNotes-style annotations converted to Stanford dependencies (de Marneffe et al., 2006). The remaining treebanks are all available in dependency format. In all cases, we apply a set of heuristics to the treebank data to make the tokenization as similar as possible to the one of the bitext. Our heuristics can split treebank tokens but do not merge treebank tokens. We found that adjusting the treebank tokenization is crucial for obtaining good results. However, this makes the reported parsing accuracies not comparable to other numbers in the literature. When necessary, we projectivize the treebanks by raising arcs until the tree becomes projective, as described in Nivre and Nilsson (2005); we do not reconstruct non-projective arcs at parsing time, since our subsequent systems expect projective trees.

Our part-of-speech tagger is a conditional random field model (Lafferty et al., 2001) with simple word-identity and affix features. The parsing model is a shift-reduce dependency parser, using the higher-order features from Zhang and Nivre (2011). Additionally, we include 256 word-cluster features (Koo et al., 2008) trained on a large amount of unlabeled monolingual text (Uszkoreit and Brants, 2008).

4 Experiments

Due to the large number of experiments and language pairs we divide the experiments into groups and discuss each in turn. We only include the results from the forest-to-string system when they are better than the phrase-based results. We use * to denote results from the forest-to-string system.

	UAS	LAS	POS
en: English[1]	92.28	90.28	97.05
cs: Czech[2]	84.66	72.01	98.97
de: German[3]	89.30	86.98	97.69
es: Spanish[4]	86.24	82.32	96.62
fr: French[5]	88.57	86.40	97.48
hu: Hungarian[2]	87.66	82.51	94.47
nl: Dutch[3]	86.09	82.31	97.38
pt: Portuguese[4]	90.22	87.26	98.10

Table 1: Parsing accuracies on the retokenized treebanks. UAS is unlabeled attachment score, LAS is labeled attachment score, and POS is part-of-speech tagging accuracy. The treebank sources are (1): Marcus et al. (1993) + Judge et al. (2006) + Petrov and McDonald (2012), (2): Nivre et al. (2007), (3): Buchholz and Marsi (2006), (4): McDonald et al. (2013), (5): Abeillé et al. (2003).

4.1 WMT Experiments

Table 2 presents detailed results on the WMT setup. Lexical reordering (Zens and Ney, 2006) never hurts and is thus included in all systems. Overall, our results are a little better than the best results of the WMT 2010 shared task for two language pairs and within reach of the best results in most other cases.

The 2-step classifier preordering approach provides statistically significant improvements over the lexical reordering baseline on three out of the eight language pairs: English-Spanish (en-es: 1.4 BLEU), German-English (de-en: 1.2 BLEU), and English-French (en-fr: 1.0 BLEU). These improvements are significant in our human side-by-side evaluation. We also observe gains when combining our preordering approach with the forest-to-string system for English-Spanish and German-English. While the forest-to-string system is capable of performing long distance reordering in the decoder, it appears that an explicitly trained lexicalized preordering model can provide complementary benefits. These benefits are especially pronounced for German-English where long distance verb movement is essential. For the romance languages (Spanish and French), word ordering depends highly on lexical choice which is captured by the lexical features in our classifiers.

	base	lexical	rule	1-step	2-step	wmt best
en-cs	14.9	**15.1**	**15.2**	**15.2**	**15.2**	15.4
en-de	15.3	15.6	**15.9**	**15.9**	**15.7**	16.3
en-es	27.4	27.8♤	28.4♧	**29.0**	**28.8**♠♣	28.6
en-es*	28.9	-	28.7	29.0	29.2	28.6
en-fr	26.3	26.5♤	26.8♧	**27.2**	**27.3**♠♧	27.6
cs-en	21.6	21.6	21.5	21.6	21.7	21.9
de-en	20.6	21.1♤	**21.9**	**21.9**	**21.8**♠	22.8
de-en*	22.1	-	**22.5**	**22.5**	**22.7**	22.8
es-en	28.3	**28.7**	**28.7**	**28.8**	**28.9**	28.8
fr-en	26.8	27.0	26.9	26.9	27.0	28.3

Table 2: BLEU scores on the WMT 2010 setup. Results from the forest-to-string system are marked with * and are only included when better than the phrase-based results. The *base* system includes a distance distortion model; the *lexical* system adds lexical reordering; *rule* is the rule preordering system of Genzel (2010) plus lexical reordering; *1-step* and *2-step* are our classifier-based systems plus lexical reordering. Bolded results are statistically significantly better than non-bolded results as measured by a bootstrap sample test with a 99% confidence interval. Human evals are conducted only where indicated; we use ♠ and ♣ to indicate a significantly better result than ♤ and ♧ in the human eval at 95%. Also included are the best results from the WMT 2010 task.

Compared to a state-of-the-art preordering system, the automatic rule extraction system of Genzel (2010), we observe significant gains in several cases and no losses at all. The improvements on English-Spanish are significant also in the human evaluation, while the English-French improvements are positive, but not statistically significant.

Comparing the different languages, Czech (cs) appears the most immune to improvements from preordering (and lexical reordering). One possible explanation is that Czech has a relatively free word order with a default SVO structure. It is therefore difficult to learn reordering changes from English to Czech. Additionally, the accuracy (LAS) of our Czech parser is by far the lowest of all parsers that we used, potentially limiting the benefits that can be obtained when translating from Czech into English.

On this setup there is fairly little difference in performance between the 1-step and 2-step approaches. The main benefit of the 2-step approach is compactness: the set of 2-step classifiers has about half the number of non-zero features as the 1-step classifiers.

4.2 Additional Languages Experiments

Table 3 shows our first set of results on the additional languages, including some languages with a wide disparity in word order relative to English. The SOV languages Korean (ko) and Japanese (ja) benefit the most from preordering and gain more than 7 BLEU relative to the phrase-based baseline and still more than 3 BLEU for the forest-to-string system. Similar improvements were reported by Xu et al. (2009) with manual reordering rules. Indonesian (id) and Malay (ms) are next with gains of 2.5 BLEU. Malay does not have a grammatical subject in the sense that English does, but instead uses a concept of an agent and an object, whose order is determined by the voice of the verb. It appears that our classifiers have learned to model some of these highly lexical, but systematic ordering preferences. Welsh (cy) and Irish (ga) as VSO languages also exhibit large gains of 2.1 BLEU. For Arabic (ar) and Hebrew (iw), the gains are smaller, but still significant and exceed 1 BLEU relative to the baseline.

The benefits of our 2-step approach over the 1-step approach become apparent on this set of languages where reordering is most important. By predicting the target word order in two steps, we reduce sparsity and make two easier decisions in place of a single difficult high entropy decision. Indeed, the 2-step approach produces improvements over the 1-step approach on five out of nine language pairs. The improvements are as large as 0.9 BLEU for Korean and 0.5 BLEU for Japanese and Welsh. We performed human evaluation for all language pairs with a noticeable BLEU gain for the 2-step system over

	base	rule	1-step	2-step
en-ar	11.4	12.3	**12.5**	**12.6**
en-cy	29.3	31.1	31.9$^{♧}$	**32.4**$^{♣}$
en-ga	17.0	18.5	18.8$^{♧}$	**19.1**$^{♣}$
en-iw	18.8	19.7	**20.2**	**20.2**
en-id	31.0	33.4	**34.0**$^{♧}$	**34.3**$^{♧}$
en-ja	10.4	16.4	17.5$^{♧}$	**18.0**$^{♣}$
en-ja*	14.9	18.0	18.2$^{♧}$	**18.6**$^{♣}$
en-ko	24.1	31.8	31.8$^{♧}$	**32.7**$^{♣}$
en-ms	20.4	22.5	**22.9**	**22.9**

Table 3: BLEU scores for language from various language families: Arabic (ar), Welsh (cy), Irish (ga), Indonesian (id), Hebrew (iw), Japanese (ja), Korean (ko), and Malay (ms). Lexical reordering is not included in any of the systems. Bolded results are significant at 99%. $^{♣}$ is significantly better than $^{♧}$ in a human eval at 95%.

the 1-step system. The human judgments exactly agree with the results of the BLEU significance tests. The gains relative to the rule reordering system of Genzel (2010) and the no-preordering baseline are even larger and therefore clearly also significant.

In Table 4 we show results for Hungarian (hu), Dutch (nl), and Portuguese (pt). In all cases but English-Hungarian we observe significant improvements over the no preordering baseline. It should be noted that the gains are not symmetric – sometimes there are larger gains for translating out of English, while for Hungarian the gains are higher for translating into English. Hungarian has a free word order which is difficult to predict which might partially explain why there are no improvements for translating into Hungarian. For Dutch-English, the forest-to-string system yields the best results, which was also the case for German-English, further supporting the observation that combining different types of syntactic reordering approaches can be beneficial.

4.3 Manually Aligned Data

	base	rule	1-step	2-step
en-hu	12.7	12.6	12.8	12.7
en-nl	25.3	26.1	**26.4**	**26.4**
en-pt	30.2	31.9	**32.6**	**32.8**
hu-en	22.0	22.2	**22.7**	**22.7**
nl-en	34.9	**35.7**	35.2	35.1
nl-en*	36.3	**36.5**	**36.6**	**36.7**
pt-en	39.8	**40.1**	**40.1**	**40.1**

Table 4: BLEU scores for translating to and from English for: Hungarian (hu), Dutch (nl), and Portuguese (pt). Lexical reordering is not used for any language pair. Bolded results are significant at 99%.

For Arabic (ar), Hebrew (iw), and Japanese (ja) we conducted some additional experiments with manually aligned data. We asked bilingual speakers to translate about 20K English sentences into the respective target language and to mark the alignment between the words. We reserved 20% of this data for evaluation and used the rest for training. For evaluation we used the fuzzy metric defined by Talbot et al. (2011), which counts the fraction of words that are reordered into the correct position.

The BLEU scores in Table 5 show that training from small amounts of manually aligned data or large amounts of automatically aligned data results in models of similar quality. In terms of the fuzzy metric, the models trained from manually aligned data were better. A possible explanation is that these models were trained on data which was much more similar to the evaluation data (both were subsets of the manually aligned data), biasing the metric in their favor. In absolute terms, the reordering accuracy is around 80% for Arabic and Japanese and close to 90% for Hebrew. Most impressively, more than 60% of the Hebrew sentences are exactly in the correct word order, implying that monotonic translation may suffice.

We also examined the accuracy of the individual classifiers and found that the pivot classifier has an accuracy around 95%. It is therefore unlikely that a word is reordered to the wrong side of its head in the 2-step reordering approach. The classifiers that predict the final word order have an accuracy above 90% when there are only two words and drop to still respectable 70%-80% when there are 4 or more children or 20 possible options.

5 Analysis

In this section, we analyze an example whose translation is significantly improved by our preordering approach, demonstrating the usefulness of our lexicalized features. Consider the English sentence:

It was a real whirlwind.

	no reordering			manual			automatic		
	fuzzy	exact	BLEU	fuzzy	exact	BLEU	fuzzy	exact	BLEU
en-ar	63.2	19.8	11.4	83.5	47.6	12.4	79.0	38.9	12.6
en-iw	67.9	22.2	18.8	89.8	62.4	20.3	89.2	61.2	20.2
en-ja*	44.1	0.0	14.9	80.9	41.5	18.4	78.5	36.8	18.6

Table 5: Preordering accuracy for the 2-step classifiers using manual alignments vs. automatic alignments. Fuzzy refers to the metric defined in Talbot et al. (2011) and exact is the percentage of sentences with a perfect preordering.

taken from the WMT test set. The dependency parse tree is shown in Figure 2. In our experiments the rule-based approach of (Genzel, 2010) reordered the source sentence into:

It was a whirlwind real.

and produced the translation:

Es un torbellino real.

In comparison, our 2-step system kept the English sentence unchanged and produced the translation:

Fue un auténtico torbellino.

The second translation is better than the first because of the correct tense (which is not related directly to the preordering) and because the noun phrase *"real whirlwind"* is ordered correctly.

The main reason for the difference in the ordering is that the rule-based system can only use the unlexicalized information from the parse tree. The head *"whirlwind"* is a noun and the child *"real"* is an adjective; since adjectives typically appear after nouns in Spanish, their order is reversed.

To understand why the classifier-based system keeps *"real"* before *"whirlwind"* we can examine the features used by the classifier to make this decision. In Table 6 we consider the 3 strongest features in favor of the child *"real"* appearing after the head *"whirlwind"* and the three strongest features in favor of the child appearing before the head. Recall that the pivot is a binary classifier: positive features support one decision (in our case: the child should be after the head) and the negative features support the other decision (the child should be before the head).

The three features that have the highest positive weight encode the fact that the child is an adjective, since in general, adjectives in Spanish appear after the noun. On the other hand, the three features with the most negative weights all encode the fact that the child is the word *"real"* which unlike most adjectives tends to appear before the noun. It is interesting to note that for this particular ordering decision the child word is much more informative than the head word and indeed, all the important features contain information about the child and none of them contains any information about the head.

6 Conclusions & Future Work

We presented a simple and novel preordering approach that produces substantial improvements in translation accuracy on a large number of languages. We use a source-side syntactic parser and train discriminative classifiers to predict the order of a parent and its children in the target language, using features from the dependency tree as well as (bi-)lexical features. To decompose the exponential space of all possible permutations, we introduce the 2-step approach. We show empirically that this approach is significantly better than directly predicting the full permutation for some languages, and never significantly worse.

We obtain strong results on the WMT 2010 shared task data, observing gains of up to 1.4 BLEU over a state-of-the-art system. We also show gains of up to 0.5 BLEU over a strong directly comparable preordering system that is based on learning unlexicalized reordering rules. We obtain improvements of more than 2 BLEU in experiments on additional languages. The gains are especially large for languages where the sentence structure is very different from English. These positive results are confirmed in human side-by-side evaluations.

When comparing our approach to syntax-based translation systems (Yamada and Knight, 2001; Galley et al., 2004; Huang et al., 2006; Dyer and Resnik, 2010) we note that both approaches use syntactic information for reordering decisions. Our preordering approach has several advantages. First, be-

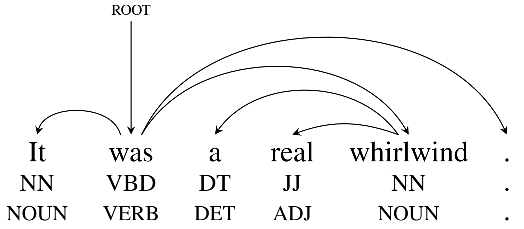

Figure 2: An example where lexical information is necessary for choosing the correct word order.

cause preordering is performed before learning word alignments, it has the potential to improve the word alignments. Second, by using discriminative classifiers we can take advantage of lexical features. Finally, preordering can be combined with syntax-based translation models and our results confirm the complementary benefits that can be obtained.

Compared to other preordering models, our approach has the obvious problem of having to make predictions over an exponential set of permutations. We show that this is not an insurmountable difficulty: our 2-step approach decomposes the exponential space, often leading to much easier prediction tasks. Even when the number of possible permutations is large we can limit ourselves to the *K* most popular permutations.

On the other hand, our approach provides important advantages. Compared to systems that use rewrite rules, it is much easier to encode useful knowledge that by itself is not enough to determine a full rewrite rule, such as “a determiner is unlikely to be the last word in a clause.” Perhaps more importantly, our model provides an elegant answer to the question of what to do when multiple rewrite rules can be applied. Previous work has employed different heuristics: use the most specific rule (Xia and McCord, 2004), use all applicable rules (Genzel, 2010), or use the most frequent rule (Wu et al., 2011). In our model there is no need for such heuristics – all the “rules” are treated as features to a discriminative classifier, and the task of analyzing their interactions is handled by the learning algorithm.

Compared to preordering systems that use ranking functions, our model has the advantage that it can encode information about the complete permutation. For example, for three source words A, B, and C, we can naturally express the useful prior that A-B-C and C-B-A are likely orders but C-A-B is not.

Feature	Weight
`PrevChild:tag=JJ,PrevSibling:a`	0.448
`PrevChild:cat=ADJ,PrevSibling:a`	0.292
`PrevChild:cat=ADJ,NoNextSibling`	0.212
...	
`PrevChild:real,NoNextHeadSibling`	-0.310
`PrevChild:real,PrevSibling:cat=DET`	-0.516
`PrevChild:real,PrevSibling:a`	-0.979

Table 6: The three features with the highest and lowest weights for choosing the position of *“real”* relative to *“whirlwind.” PrevChild* means that the child is the immediate word before the head. *PrevSibling* refers to the child’s sibling immediately to the left (the determiner *“a”*). *NoNextSibling* and *NoNextHeadSibling* mean that the child and head do not have a sibling to the right.

Promising directions for future work are joint parsing and reordering models, and measuring the influence of parsing accuracy on preordering and final translation quality.

References

A. Abeillé, L. Clément, and F. Toussenel. 2003. Building a Treebank for French. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, chapter 10. Kluwer.

P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2).

P. F. Brown, V. J. Della Pietra, S. A. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19.

S. Buchholz and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proc. of CoNLL ’06*.

C. Callison-Burch, P. Koehn, C. Monz, K. Peterson, M. Przybocki, and O. Zaidan. 2010. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proc. of ACL’05 WMT*.

M. Collins, P. Koehn, and I. Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proc. of ACL ’05*.

M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *ACL ’97*.

M.-C. de Marneffe, B. MacCartney, and C. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. of LREC ’06*.

J. DeNero and J. Uszkoreit. 2011. Inducing sentence structure from parallel corpora for reordering. In *Proc. of EMNLP '11*.

J. Duchi and Y. Singer. 2009. Boosting with structural sparsity. In *Proc. of ICML '09*.

C. Dyer and P. Resnik. 2010. Context-free reordering, finite-state translation. In *Proc. of NAACL-HLT '10*.

M. Galley, M. Hopkins, K. Knight, and D. Marcu. 2004. What's in a translation rule? In *Proc. of NAACL-HLT '04*.

D. Genzel. 2010. Automatically learning source-side reordering rules for large scale machine translation. In *Proc. of COLING '10*.

N. Habash. 2007. Syntactic preprocessing for statistical machine translation. In *Proc. of MTS '07*.

L. Huang, K. Knight, and A. Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. of AMTA '06*.

J. Judge, A. Cahill, and J. v. Genabith. 2006. QuestionBank: creating a corpus of parse-annotated questions. In *Proc. of ACL '06*.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase based translation. In *Proc. of NAACL-HLT '03*.

T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proc. of ACL-HLT '08*.

S. Kumar, W. Macherey, C. Dyer, and F. Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proc. of ACL '09*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML '01*.

C. H. Li, M. Li, D. Zhang, M. Li, M. Zhou, and Y. Guan. 2007. A Probabilistic Approach to Syntax-based Reordering for Statistical Machine Translation. In *Proc. of ACL '07*.

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. In *Computational Linguistics*.

R. McDonald, J. Nivre, Y. Quirmbach-Brundagez, Y. Goldberg, D. Das, K. Ganchev, K. Hall, S. Petrov, H. Zhang, O. Täckström, C. Bedini, N. Bertomeu Castelló, and J. Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proc. of ACL '13*.

G. Neubig, T. Watanabe, and S. Mori. 2012. Inducing a discriminative parser to optimize machine translation reordering. In *Proc. of EMNLP-CoNLL '12*.

J. Nivre and J. Nilsson. 2005. Pseudo-projective dependency parsing. In *Proc. of ACL '05*.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proc. EMNLP-CoNLL '07*.

F. J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4).

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL '02*.

S. Petrov and R. McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Proc. of NAACL '12 SANCL*.

S. Petrov, D. Das, and R. McDonald. 2012. A universal part-of-speech tagset. In *Proc. of LREC '12*.

D. Talbot, H. Kazawa, H. Ichikawa, J. Katz-Brown, M. Seno, and F. Och. 2011. A lightweight evaluation framework for machine translation reordering. In *Proc. of EMNLP '11 WMT*.

C. Tillmann. 2004. A unigram orientation model for statistical machine translation. In *Proc. of NAACL-HLT '04*.

R. Tromble and J. Eisner. 2009. Learning linear ordering problems for better translation. In *Proc. of EMNLP '09*.

J. Uszkoreit and T. Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *Proc. of ACL-HLT '08*.

S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *In Proc. of COLING '96*.

C. Wang, M. Collins, and P. Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proc. of EMNLP-CoNLL '07*.

X. Wu, K. Sudoh, K. Duh, H. Tsukada, and M. Nagata. 2011. Extracting pre-ordering rules from predicate-argument structures. In *Proc. of IJCNLP '11*.

F. Xia and M. McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proc. of COLING '04*.

P. Xu, J. Kang, M. Ringgaard, and F. Och. 2009. Using a dependency parser to improve SMT for subject-object-verb languages. In *Proc. of NAACL-HLT '09*.

K. Yamada and K. Knight. 2001. A syntax-based statistical translation model. In *Proc. of ACL '01*.

N. Yang, M. Li, D. Zhang, and N. Yu. 2012. A ranking-based approach to word reordering for statistical machine translation. In *Proc. of ACL '12*.

R. Zens and H. Ney. 2006. Discriminative reordering models for statistical machine translation. In *Proc. of NAACL '06 WMT*.

Y. Zhang and J. Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proc. of ACL-HLT '11*.

H. Zhang, L. Fang, P. Xu, and X. Wu. 2011. Binarized forest to string translation. In *Proc. of ACL-HLT '11*.