Dual-Path Phrase-Based Statistical Machine Translation

Susan Howlett and **Mark Dras**
Centre for Language Technology
Macquarie University
Sydney, Australia
{susan.howlett,mark.dras}@mq.edu.au

Abstract

Preceding a phrase-based statistical machine translation (PSMT) system by a syntactically-informed reordering preprocessing step has been shown to improve overall translation performance compared to a baseline PSMT system. However, the improvement is not seen for every sentence. We use a lattice input to a PSMT system in order to translate simultaneously across both original and reordered versions of a sentence, and include a number of confidence features to support the system in choosing on a sentence-by-sentence basis whether to use the reordering process. In German-to-English translation, our best system achieves a BLEU score of 21.39, an improvement of 0.62.

1 Introduction

Machine translation (MT) is the automatic translation of text from one human language to another. Statistical MT accomplishes this through a probabilistic model of the translation process.

In phrase-based statistical MT (PSMT), translation proceeds by dividing a sentence into sequences of adjacent words called *phrases*, then translating each phrase and reordering the phrases according to a *distortion model*. The distortion model may be lexicalised but does not typically incorporate information about the syntactic structure of the sentence. As such, although PSMT has been very successful, it suffers from the lack of a principled mechanism for handling long-distance reordering phenomena due to word order differences between languages.

One method for addressing this difficulty is the *reordering-as-preprocessing* approach, exemplified by Collins et al. (2005) and Xia and McCord (2004), where PSMT is coupled with a preprocessing step that reorders input sentences to more closely parallel the target language word order. Although this leads to improved performance overall, Collins et al. (2005) show that the reordering-as-preprocessing system does not consistently provide better translations than the PSMT baseline on a sentence-by-sentence basis.

One possible reason could be errors in the parse or the consequent reordering. Chiang et al. (2009) used features indicating problematic use of syntax to improve performance within hierarchical and syntax-based translation. In this work, we want to see whether syntax-related features can help choose between original and reordered sentence translations in PSMT.

We use as our starting point the PSMT system Moses (Koehn et al., 2007). In order to use features within the system's log-linear model to assess the reliability of syntax, it is necessary to input both variants simultaneously. To do this, we adapt in a novel way the *lattice input* of Moses; we refer to this new system as *dual-path PSMT* (§3). We then augment the model with a number of *confidence features* to enable it to evaluate which of the two paths is more likely to yield the best translation (§3.2). We reimplement the Collins et al. (2005) reordering preprocessing step and conduct some preliminary experiments in German-to-English translation (§4).

Our results (§5) do not replicate the finding of Collins et al. (2005) that the preprocessing step produces better translation results overall. However, results for our dual-path PSMT system do show an improvement, with our plain system achieving a BLEU score (Papineni et al., 2002) of 21.39, an increase of 0.62 over the baseline. We therefore conclude that a syntactically-informed reordering preprocessing step is inconsistently of use in PSMT, and that enabling the system to choose when to use the reordering leads to improved translation performance.

Susan Howlett and Mark Dras. 2010. Dual-Path Phrase-Based Statistical Machine Translation. In *Proceedings of Australasian Language Technology Association Workshop*, pages 32−40

2 Background and Related Work

2.1 Phrase-based statistical MT

Phrase-based statistical MT (PSMT) systems such as Marcu and Wong (2002) and Koehn et al. (2007) have set the standard for statistical MT for many years. While successful, these systems make limited use of linguistic information about the syntax of the languages which, intuitively, would seem to be useful. Much research is now focused on how to incorporate syntax into statistical MT, for example by using linguistic parse trees on one or both sides of the translation (e.g. Yamada and Knight (2001), Quirk et al. (2005)) or by incorporating only select aspects of syntactic structure, such as recursive structure (Chiang, 2007) or discontinuous phrases (Galley and Manning, 2010).

One area where this lack of syntactic information is felt is the *distortion model* of the PSMT system. This component governs the relative movement of phrases and is the primary means of dealing with word order differences in translation. Common options are distance-based models (where movement is penalised proportionally to the distance moved) and lexicalised models (where probability of movement is conditioned upon the phrase being moved). Without syntactic information here, PSMT systems lack a principled way to manage long-distance movements, leading to difficulty in language pairs where this is needed, such as English and Japanese or German.

2.2 Reordering-as-preprocessing

The *reordering-as-preprocessing* approach addresses the PSMT reordering difficulty by removing word order differences prior to translation. This is done with a preprocessing step where the input sentence is parsed and a reordered alternative created on the basis of the resulting parse tree.

Our work builds on the reordering-as-preprocessing approach of Collins et al. (2005). Working with German-to-English translation, Collins et al. (2005) parse input sentences with a constituent-structure parser and apply six hand-crafted rules to reorder the German text toward English word order. These rules target the placement of non-finite and finite verbs, subjects, particles and negation. The authors demonstrate a statistically significant improvement in BLEU score over the baseline PSMT system.

Many other reordering-as-preprocessing systems exist. Xia and McCord (2004) present a system for French–English translation that, instead of using hand-crafted reordering rules, automatically learns reordering patterns from the corpus. Automatically-acquired rules may be noisier and less intuitive than hand-crafted rules but the approach has the advantage of being more easily extended to new language pairs. Other examples of systems include Wang et al. (2007) (manual rules, Chinese-to-English), Habash (2007) (automatic rules, Arabic-to-English) and Popović and Ney (2006) (manual rules, Spanish/English-to-Spanish/English/German).

Despite the success of the reordering-as-preprocessing approach overall, Collins et al. (2005) found that in a human evaluation on 100 sentences, there were still several cases in which the baseline system translation was preferred over that produced with the reordering. The authors note this finding but do not analyse it further.

2.3 Features for improved translation

Zwarts and Dras (2008) explore the Collins et al. (2005) finding by examining whether machine learning techniques can be used to predict, on a sentence-by-sentence basis, whether the translation of the reordered sentence is to be preferred over the alternative. For features, they use sentence length, parse probability from the Collins parser and unlinked fragment count from the Link Grammar parser on the English side of the translation. The authors find that, when used on the source side (in English-to-Dutch translation), these features provide no significant improvement in BLEU score, while as target-side features (in Dutch-to-English translation) they improve the BLEU score by 1.7 points over and above the 1.3 point improvement from reordering.

Our work has some similarities to that of Zwarts and Dras (2008) but uses the log-linear model of the translation system itself to include features, rather than a separate classifier that does not permit interaction between the confidence features and features used during translation.

This idea of using linguistic features to improve statistical MT has appeared in a number of recent papers. Chiang et al. (2009) demonstrate an improvement in hierarchical PSMT and syntax-based (string-to-tree) statistical MT through the addition of features pinpointing possible errors in the translation, for example the number of occurrences of a particular grammar production rule, or

non-terminal in a rule. Xiong et al. (2010) derive features from the Link Grammar parser, in combination with word posterior probabilities, to detect MT errors (in order to subsequently improve translation quality). Unlike Chiang et al. (2009), we work with PSMT and use features that consider the parse tree as a whole or aspects of the reordering process itself. Unlike Xiong et al. (2010), we use these features directly in translation.

2.4 Reordering lattices

Our work also bears some similarity to recent work using a *reordering lattice* or *reordering forest* as input to the translation system. Examples include Ge (2010), Dyer and Resnik (2010) and Zhang et al. (2007). In these systems, the input structure simultaneously represents many possible reorderings of a sentence, which are produced from a single parse and capture all possible combinations of individual reordering choices.

Like these systems, we use a complex input structure to translate across multiple variations of the sentence simultaneously, and choose between the resulting translations within the translation model itself. However, like Zwarts and Dras (2008), we consider only two possibilities: the original sentence and the sentence with a particular reordering process applied in full.

Our work also differs from these lattice-based systems in that we preserve and incorporate the standard distortion model of the PSMT system in a way that the above systems cannot. Where the lattice-based systems aim to overcome the weaknesses of the distortion model by replacing it with the lattice-creating reordering process, we view the two components as complementary. This has an interesting consequence, which we introduce in §3.1 and discuss further in §5.

Similar to our work and the work in the previous section, Ge (2010) includes features to assess reordering options, based on the structure of the resulting tree, for example which nonterminal appears as the first child and the size of jump required to reach the nonterminal used as the next child. In addition to the difference with the distortion model mentioned above, our work differs in that Ge (2010) focuses on finding the best reordering using syntactic features plus a few surface and POS-tag features as a way of "guarding against parsing errors", whereas we also look at using features to represent confidence in a parse.

3 Dual-Path PSMT

In this paper, we develop a *dual-path* PSMT system. §3.1 introduces the lattice input format, by which we provide the system with two variants of the input sentence: the original and the reordered alternative produced by the preprocessing step. §3.2 outlines the confidence features that we include in the translation model to help the system choose between the two alternatives.

Our system is built upon the PSMT system Moses (Koehn et al., 2007). For reordering, we use the Berkeley parser (Petrov et al., 2006) and the rules given by Collins et al. (2005), but any reordering preprocessing step could equally be used. Further details of our systems are given in §4.

3.1 Word Lattices

A *word lattice* is a structure used to efficiently represent multiple sentences simultaneously. This is of use, for instance, in translating the output of a speech recognition system, where there is some uncertainty about the words of the sentence. The lattice is a directed, acyclic graph with one source and one sink node. Each path from source to sink corresponds to one of the set of sentences being represented. An example is given in Figure 1.

There exists an approximation to the word lattice structure, called a *confusion network*. In this case, every path of the lattice passes through every node, and epsilon transitions are allowed. While a confusion network is in general more space-efficient than a lattice, its paths may include more sentences than the original set (Koehn, 2010).

In the word lattice, each edge is accompanied by a *transition probability*; in speech recognition output this represents the probability that the edge is the correct interpretation of the next part of the signal, with the probabilities on all of the transitions out of a single node summing to one. The probability of one path in the lattice (and therefore one sentence in the set) is the product of the probabilities on the transitions that make up the path. This path probability becomes one feature in the system's translation model.

Moses can be used with word lattice input. Typically, training proceeds as in the baseline case, extracting and scoring phrase pairs from plain parallel sentence data. The lattice structure first appears in tuning, where Moses determines the weight to assign to the transition probability feature, which is then used to translate lattices during evaluation.

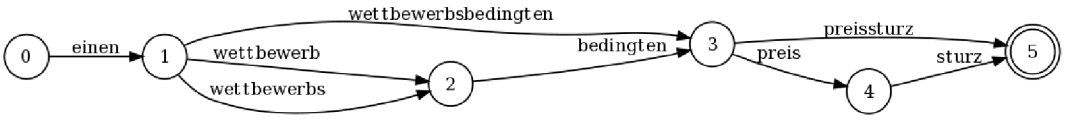

Figure 1: A example of a word lattice for decomposing German compounds, from Koehn (2010).

Our system In our dual-path system, we use a word lattice to simultaneously represent original and reordered versions of an input sentence. Representing only two versions of the sentence, our lattice contains only two paths. We do not wish to introduce any additional paths that are combinations of these two, since doing so sensibly is nontrivial, so we do not use the confusion network approximation.

As mentioned in §3, we use our lattice input in conjunction with the PSMT distortion model. Commonly, Moses is used with a lexicalised distortion model, which is trained along with the phrase table from the training data. In that case, the distortion model for the original sentences will not be appropriate for the reordered sentences, and vice versa. We therefore need separate distortion information for the two paths of the lattice. To accomplish this, we use disjoint vocabularies on the two paths, which we create by prepending every token in the original sentences with `1_` and every token in the reordered sentences with `2_`.

This has two consequences. First, since Moses translates unknown vocabulary items by copying them directly to the output, we must ensure that the prefix is removed before this copying occurs. Second, by introducing disjoint vocabularies for the two paths, we inhibit any sharing of edges between them. This in turn inhibits any sharing of nodes since, like in confusion networks, this would increase the number of paths. Our lattice is therefore degenerate, with the two paths sharing only the source and sink nodes. An example of our lattice input is shown in Figure 2.

This complete separation of the two paths means that extending the dual-path system to a multi-path system is likely to rapidly encounter efficiency issues. It is, however, a possibility; we discuss a number of options in §5.

Finally, when specifying the lattice, nodes must be given in topographical order. We number the nodes of each path alternately since for efficiency reasons Moses imposes a limit on the length of edges in the lattice.

Note about training As in the typical case, our lattice is used only during tuning and evaluation. For training, we construct a parallel corpus twice the size of the original corpus, containing the original sentences (prefixed with `1_`) plus their translations, and the reordered versions of these sentences (prefixed with `2_`) plus their translations (identical to those of the first half). Training takes place on this one corpus, giving one phrase table containing translation candidates for both paths.

Using one vocabulary The need for disjoint vocabularies is a consequence of the lexicalised distortion model; with a simple distance-based model this would not be necessary. In that case, we could construct a lattice identical to Figure 2 but with the `1_/2_` prefixes removed (which may then allow some compression of the lattice), and then proceed as in the disjoint-vocabulary system. In this case, it may also be necessary to create separate phrase tables from the two halves of the training data.

3.2 Confidence Features

When translating lattice input, Moses creates translation candidates for all (in our case, both) paths in the lattice, and the ultimate translation will be one option from one path. The probability of a path (obtained by multiplying the probabilities of its component edges) is used as a feature for every translation candidate for that path. We can therefore use the transition probabilities to influence the system to choose the best translation of one path over the best translation of the other path.

Usefully, Moses treats the transition probabilities like any other feature, so in fact the values on the transitions need not fall in the range $[0, 1]$ and the values on transitions out of a single node need not sum to one. Further, Moses allows multiple such features, so each edge in the lattice may be associated with n values and thus each path may have n scores, each produced by multiplying the corresponding values on its component edges.[1]

[1]Moses actually operates with log-probabilities. After the lattice is read in, a log transformation is applied to every value

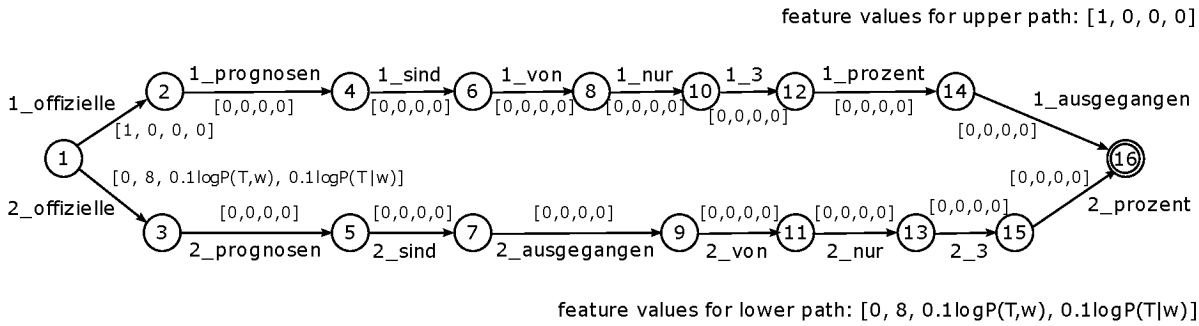

Figure 2: Example input to our dual-path system, illustrating disjoint vocabularies, node ordering and four feature values (both per edge and combined for path totals), for the sentence *Offizielle Prognosen sind von nur 3 Prozent ausgegangen* ('Official forecasts predicted just 3 percent'), adapted from the tuning data.

We first outline the features that we use, and then discuss how they are incorporated into the translation model.

3.2.1 Feature set

Reordering indicator feature This feature is intended to differentiate the two paths of the lattice. Its function is described more fully in §3.2.2.

Sentence length Intuitively, longer sentences are more likely to be poorly parsed or poorly reordered, so as a feature we include the number of tokens in the original sentence.

Parse probability (2 features) The reordering is based on an automatically-produced parse. A low parse probability may indicate that the sentence was difficult to parse and the reordering may therefore be unreliable. To capture this, we include features based on the joint and conditional probabilities of the parse tree: $P(T, w)$ and $P(T|w)$. Both are included as it is not obvious whether one or the other will be the more useful.

The Berkeley parser can provide both $\log P(T, w)$ and $\log P(T|w)$ with the parse of the sentence. To avoid underflow issues, we actually use $0.1 \times \log P(T, w)$ and $0.1 \times \log P(T|w)$.

Rule application counts (6 features) We include counts of the number of times each of the six Collins et al. (2005) reordering rules was used for the sentence, since particular rules may be more likely to cause problems in the reordering. We anticipate this feature would be more useful in a system with automatically-acquired reordering rules.

on every edge. In the discussion that follows and in Figure 2 we specify the feature values as those obtained after the log transformation.

POS counts (4 features) These features count the number of nouns, verbs, prepositions and conjunctions in the sentence. Higher counts may indicate more complex sentences that require more reordering or are more likely to be poorly reordered.

Jump sizes (10 features) Each time a reordering rule fires, a sequence of a tokens is moved past a sequence of b tokens. We call this a jump of size $\max(a, b)$ (as it may be either a sequence of a tokens jumping over a sequence of b tokens, or vice versa). These jump sizes are binned, and the number of jumps observed in each bin for the sentence becomes one feature in the model.

To choose the bins, we examined the jumps that occur when reordering the smaller of our parallel training corpora.[2] (Recall that the lattices and therefore these confidence features do not appear in the system until the tuning stage.) We observed jumps from 1 to 94 tokens in size, with the number of jumps observed decreasing roughly exponentially as the size of the jump increased.

Although the reordering process is intended to produce long jumps, we expect that jumps above a certain size are more likely to be due to erroneous bracketing in the parse. However, we are not sure at which point a jump becomes too large, and we may wish to apply progressively stronger penalties for longer jumps. We therefore prefer a larger number of narrower bins over fewer, wider bins. We chose the bins listed in Table 1.

Table 1 also lists the number of times a jump of this size was seen in the training set, and the number of times we therefore expect to see a jump of this size in the tuning set, where the weights for each bin feature will be set.

[2] news-commentary10.de-en.de

Bin	# in training	Expected # in tuning
1–2	104,296	2,133
3–5	89,383	1,828
6–8	46,695	955
9–12	27,484	562
13–16	10,935	234
17–20	4,739	97
21–25	2,610	53
26–30	1,115	23
31–35	498	10
36+	455	9

Table 1: Jumps seen in the training set of 100,269 sentence pairs, and the expected number for each bin in the tuning set of 2,051 sentence pairs, if the two corpora have a similar distribution.

3.2.2 Inclusion in the translation model

Recall that our input lattice consists of two disjoint paths. The only choice occurs at the first node of the lattice, where there are two options; all other nodes have precisely one outgoing transition.

Of the two transitions out of the first node of the lattice, one is for the original sentence and the other for the reordered variant. On the original sentence transition, we put the value 1 for the indicator feature and 0 for the remaining 23 confidence features. On the reordered sentence transition, we put 0 for the indicator feature and for all other features the value as specified in §3.2.1.

For all the other transitions in the lattice, where there is no branching, we put the value 0 for all 24 confidence features. This means that these edges do not contribute to the scores for each path, and so the overall feature values for each path are the same as those of its first edge. The numbers beside each edge in Figure 2 demonstrate the result with the first four features. Note that all translation candidates for a given path will share the same feature values; the features distinguish between paths, not between candidates from one path.

Moses uses a maximum entropy model to score translation candidates. As such, the final score of a translation candidate is the weighted sum of all of the feature values of the translation:

$$\text{score} = \exp \sum_{i=1}^{n} \lambda_i h_i(e, f) \qquad (1)$$

This automatically includes the lattice transition features. Assume without loss of generality that the confidence features above are features $1, \ldots, m$, with feature 1 being the reordering indicator feature. Therefore the part of this sum that is due to the confidence features will be $\sum_{i=1}^{m} \lambda_i h_i(e, f)$. This will simplify to λ_1 for candidate translations of the original sentence and $\sum_{i=2}^{m} \lambda_i h_i(e, f)$ for candidate translations of the reordered sentence. Hence we expect the features excluding the indicator feature to collectively indicate the extent to which the reordering may be trusted, with λ_1 controlling the general preference to use the original or the reordered sentence.[3]

The current version of Moses uses minimum error rate training (MERT) to set the weights λ_i. This procedure is limited in the number of features that it can efficiently process. Further extending the feature set would require a different optimisation procedure, such as MIRA, as discussed by Arun and Koehn (2007).

4 Experiments

4.1 Models and Evaluation

Our baseline PSMT system is Moses (Koehn et al., 2007), repository revision 3590.[4] We run all of our experiments using the Moses Experiment Management System; configuration files and scripts to reproduce our experiments are available online.[5]

For the reordering preprocessing step we reimplement the Collins et al. (2005) rules and use this to recreate the Collins et al. (2005) reordering-as-preprocessing system as our second baseline.

We use the Berkeley parser (Petrov et al., 2006), repository revision 14,[6] to provide the parse trees for the reordering process. Since the German parsing model provided on the parser website does not include the function labels needed by the Collins et al. (2005) rules, we trained a new parsing model on the Tiger corpus (version 1). The reordering script and parsing model, along with details of how the parsing model was trained, are available online with the configuration files above.

We compare four systems on German-to-English translation: the Moses baseline (MOSES), the Collins et al. (2005) baseline (REORDER), the lattice system with just the reordering indicator feature (LATTICE), and the lattice system with all

[3] It is possible that in practice the imbalance in number of non-zero features between the two paths could cause the system some difficulty in assigning the weights for each feature. In future it would be interesting to investigate this possibility by introducing extra features to balance the two paths.

[4] https://mosesdecoder.svn.sourceforge.net/svnroot/mosesdecoder/trunk/

[5] http://www.showlett.id.au/. Some minor changes to Moses were implemented to work with our job scheduling software; full details are also available here.

[6] http://berkeleyparser.googlecode.com/svn/trunk/

Purpose	File set	# Sents
Training	europarl-v5.de-en	1,540,549
	news-commentary10.de-en	100,269
Tuning	news-test2008	2,051
Test	newstest2009	2,525
LM	europarl-v5.en	1,843,035
	news-commentary10.en	125,879
	news.en.shuffled	48,653,884

Table 2: Corpora used in our experiments and their sizes. The first four are parallel corpora (size: number of sentence pairs); the last three are monolingual corpora for the language model (size: number of sentences).

confidence features (+FEATURES). We do not explore different subsets of the features here.

For evaluation we use the standard BLEU metric (Papineni et al., 2002), which measures n-gram overlap between the candidate translation and the given reference translation.

4.2 Approximate Oracle

To get an idea of a rough upper bound, we implement the approximate oracle outlined in Zwarts and Dras (2008). For every sentence, the approximate oracle compares the outputs of MOSES and REORDER with the reference translation and chooses the output that it expects will contribute to a higher BLEU score overall. The oracle chooses the candidate that shares the highest number of 4-grams with the reference; if the two have the same 4-gram overlap, it chooses the one that shares the highest number of trigrams with the reference, and so on down to unigrams. If still identical, it chooses the original.

Note that the output of the oracle for some sentence will be identical to the output of one or the other baseline system; for the lattice system this is not necessarily the case since the system is tuned separately with a different number of features.

4.3 Data

For data we use the corpora provided for the 2010 Workshop on Statistical Machine Translation[7] translation task. The number of sentence pairs in each corpus are given in Table 2.

We trained 5-gram language models with SRILM (Stolcke, 2002) using the three language model files listed in Table 2. For convenience, the news.en.shuffled corpus was split into eight smaller files, seven containing 6,100,000 lines and the last containing the remainder. One language model was produced for each file or subfile, giving a total of ten models. The final language model was produced by interpolation between these ten, with weights assigned based on the tuning corpus.

[7] http://www.statmt.org/wmt10/

System	BLEU
MOSES	20.77
REORDER	20.04
Approx oracle	22.45
LATTICE	21.39
+FEATURES	21.10

Table 3: BLEU scores for every system

5 Results and Discussion

Table 3 gives the BLEU score for each of our four systems and the approximate oracle. We note that these numbers are lower than those reported by Collins et al. (2005). However, this is most likely due to differences in the training and testing data; our results are roughly in line with the numbers reported in the Euromatrix project for this test set.[8]

Interestingly, our reimplementation of the Collins et al. (2005) baseline does not outperform the plain PSMT baseline. Possible explanations include variability due to differences in training data, noisier parser output in our system, or differing interpretation of the description of the reordering rules. It may also be that the inconsistency of improvement noted by Collins et al. (2005) is the cause; sometimes the reordering produces better results and sometimes the baseline, with the effect just by chance favouring the baseline here. To explore this, we look at the approximate oracle.

In our experiment, the oracle preferred the baseline output in 848 cases and the reordered in 1,070 cases. 215 sentences were identical between the two systems, while in 392 cases the sentences differed but had equal numbers of n-gram overlaps. The BLEU score for the oracle is higher than that of both baselines; from this and the distribution of the oracle's choices, we conclude that the difference between our findings and those of Collins et al. (2005) is at least partly due to the inconsistency that they identified. It is especially interesting to note that the reordered system's translations are preferred by the oracle more often even though its overall performance is lower.

Turning now to the results of our systems, we see that simply giving the system the option of

[8] http://matrix.statmt.org/

both reordered and non-reordered versions of the sentence (LATTICE) produces an improved translation performance overall. While the addition of our confidence features (+FEATURES) leaves the performance roughly unchanged, the gap between LATTICE and the approximate oracle implies that this is due to the choice of features, and that a feature set may yet be found that will improve performance over the plain LATTICE system.

In light of the Zwarts and Dras (2008) finding that source-side features in the classifier do not help translation performance, our negative result with +FEATURES may appear unsurprising, as all of our features may be classified as source-side. However, we note that there remains a considerable feature space to explore. Note that if we were to include the equivalent of their target-side features into our system, they would appear as language model features, rather than features on the input lattice. Thus it seems that in fact Zwarts and Dras (2008) address two distinct problems—adding syntactic information to the translation process, and adding it to the language model.

5.1 Extensions and Future Work

Given our dual-path system, an obvious question is whether a multi-path extension is possible. Since the disjoint vocabularies inhibit compression of the lattice, extending the input to a multi-path lattice is likely to rapidly encounter efficiency issues. However, some possibilities exist.

One option would be to include paths that represent different reorderings of the sentence based on the same parse. For example, the original sentence could be compared with reorderings created by different reordering-as-preprocessing approaches. In this instance, it would be advisable to use a different vocabulary (by using a different token prefix) for each path, as each reordering is likely to require a different lexicalised distortion model.

In the case where these reordered alternatives are all possible combinations of parts of one reordering process, our system approaches the work described in §2.4, and in fact those systems will probably be more suitable as the preprocessing takes over the role of the PSMT distortion model.

Alternatively, the multiple options could be created by the same preprocessor but based on different parses, say the n best parses returned by one parser, or the output of n different parsers with comparable outputs. This extension would be quite different from the lattice-based systems in §2.4, which are all based on a single parse.

For future systems, we would like to replace the Collins et al. (2005) reordering rules with a set of automatically-extracted reordering rules (as in Xia and McCord (2004)) so that we may more easily explore the usefulness of our system and confidence features in new language pairs with a variety of reordering requirements.

The next major phase of this work is to extend and explore the feature space. This entails examining subsets of confidence features to establish which are the most useful indicators of reliable reordering, and possibly replacing the MERT tuning process with another algorithm, such as MIRA, to handle a greater quantity of features. In addition, we wish to explore more fully our negative result with the reimplementation of the Collins et al. (2005) system, to investigate the effect of balancing features in the lattice, and to examine the variability of the BLEU scores for each system.

6 Conclusion

We adapt the lattice input to the PSMT system Moses to create a system that can simultaneously translate a sentence and its reordered variant produced by a syntactically-informed preprocessing step. We find that providing the system with this choice results in improved translation performance, achieving a BLEU score of 21.39, 0.62 higher than the baseline.

We then augment the translation model of our system with a number of features to express our confidence in the reordering. While these features do not yield further improvement, a rough upper bound provided by our approximate oracle suggests that other features may still be found to guide the system in choosing whether or not to use the syntactically-informed reordering.

While our reordering step is a reimplementation of the Collins et al. (2005) system, contrary to their findings we do not see an improvement using the reordering step alone. This provides evidence against the idea that reordering improves translation performance absolutely. However, our success with the lattice system highlights the fact that it *is* useful for some sentences, and that syntactic confidence features may provide a mechanism for identifying which sentences, thus incorporating syntactic information into phrase-based statistical machine translation in a useful way.

Acknowledgements

We would like to thank the anonymous reviewers, Chris Dyer and Mark Johnson for their helpful comments, and Christian Hardmeier and Simon Zwarts for implementation assistance.

References

Abhishek Arun and Philipp Koehn. 2007. Online learning methods for discriminative training of phrase based statistical machine translation. In *Proceedings of the MT Summit XI*, pages 15–20.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 218–226.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 531–540.

Chris Dyer and Philip Resnik. 2010. Context-free reordering, finite-state translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 858–866.

Michel Galley and Christopher D. Manning. 2010. Accurate non-hierarchical phrase-based translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 966–974.

Niyu Ge. 2010. A direct syntax-driven reordering model for phrase-based machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 849–857.

Nizar Habash. 2007. Syntactic preprocessing for statistical machine translation. In *Proceedings of the MT Summit XI*, pages 215–222.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180.

Philipp Koehn, 2010. *Moses: Statistical Machine Translation System*. University of Edinburgh. http://www.statmt.org/moses/manual/manual.pdf Downloaded 13 July 2010.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–139.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440.

Maja Popović and Hermann Ney. 2006. POS-based word reorderings for statistical machine translation. In *Proceedings of LREC 2006*, pages 1278–1283.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 271–279.

Andreas Stolcke. 2002. Srilm — an extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, pages 901–904.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 737–745.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 508–514.

Deyi Xiong, Min Zhang, and Haizhou Li. 2010. Error detection for statistical machine translation using linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 604–611.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530.

Yuqi Zhang, Richard Zens, and Hermann Ney. 2007. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 1–8.

Simon Zwarts and Mark Dras. 2008. Choosing the right translation: A syntactically informed classification approach. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 1153–1160.