Syntax-Based Word Reordering in Phrase-Based Statistical Machine Translation: Why Does it Work?

Simon Zwarts
Centre for Language Technology
Macquarie University
Sydney, Australia
szwarts@ics.mq.edu.au

Mark Dras
Centre for Language Technology
Macquarie University
Sydney, Australia
madras@ics.mq.edu.au

Abstract

Most natural language applications have some degree of preprocessing of data: tokenisation, stemming and so on. In the domain of Statistical Machine Translation (SMT) it has been shown that word reordering as a preprocessing step can help the translation process, but it is unclear why. We propose two possible reasons for the observed improvement: (1) that the reordering explicitly matches the syntax of the source language more closely to that of the target language; or (2) that it fits the data better to the mechanisms of phrasal SMT. In previous work from German to English, for example, hand-written language-specific reordering rules both match the German more closely to English syntax, and compress heads and dependants into the PSMT phrasal window. Whether the source of the improvement is (1) or (2) has not been determined, although most other work assumes the former.

To identify the effects of each possible cause, we carry out two sets of experiments. For (1) we reverse the language-dependent syntactic reordering such that heads and dependants are moved apart. For (2), we propose a generic approach to minimising dependency distances in reordering that does not explicitly match target language word order and that does not require language-specific rules; the aim of which, rather than to beat state-of-the-art systems, is to investigate. The results show that (1) and (2) individually do still lead to improvements in translation quality, but each weaker than the original, suggesting that both features are necessary for a strong improvement. A consequence of this is that is possible to gain half the improvement of language-specific rules through one generic one.

Much to learn, you still have
Jedi Master Yoda

1 Introduction

Preprocessing—tokenisation, stemming and so on—is an essential step in natural language applications. Reordering of words on a sentence level as a more extensive step for preprocessing has succeeded in improving results in Statistical Machine Translation (SMT). Here, in both the training and decoding phases, sentences in the source language are reordered before being processed.

Although in the Machine Translation (MT) community it is still an open question, syntactic information seems to be able to help in SMT when applied correctly. An example of this is the work of Quirk et al. (2005) where a dependency parser was used to learn certain translation phrases, in their work on 'treelets'. Also Marcu et al. (2006) present a syntax-based approach with phrases that achieves a convincing quality improvement over phrases without these syntax rules. In this paper, however, we take as a starting point approaches which use syntax in a preprocessing step.

Word reordering can be done based on rules over word alignment learnt statistically, for example Costa-Jussà and Fonollosa (2006). In this work an improvement in overall translation quality in a Spanish-English MT system was achieved by using statistical word classes and a word-based distortion model to reorder words in the source language. Reordering here is purely a statistical process and no syntactic knowledge of the language is used.

Xia and McCord (2004) do use syntactic knowledge; they use pattern learning in their reordering system. In their work they parse and align sentences in the training phase and derive reordering patterns. From the English-French Canadian Hansard they extract 56,000 different transformations for translation. In the decoding phase they use these transformations on the source language. The main focus then is monotonic decoding. Both of these two cited works assume that explicitly matching the word order of the target language is the key.

Syntactically motivated rules based on clause restructuring are also used in reordering models. In Collins et al. (2005) six hand-written rules for reordering source sentences are defined. These rules operate on the output of an automatic parser. The reordering rules however are language-pair (German-English) specific and hand-written.

Syntactical reordering has also shown to improve machine translation quality for translation outside European languages, that is from an Indo-European language (English) to an Asian language (Chinese) (Wang et al., 2007). This shows the importance of reordering models as preprocessing in SMT, and reinforces the question as to why reordering works.

We note that a common characteristic of the Collins et al. (2005) rules is that they reduce the distances of a certain class of long-distance dependencies in German with respect to English, which is known to have quite short distances between heads and dependants. Thus this approach has the effect of reducing semantically linked elements into the same phrasal window, in addition to matching the word order of English. To tease out the effects of these two possible causes, we carry out two sets of experiments, us-

ing Dutch and English as our languages. First we replicate Collins et al. work for Dutch to English, just to use as our starting point. Then to test the effect of matching word order only, we reverse the long distance reordering rules, consequently moving heads and dependants apart while matching word order. To reverse the distance we switch language pair. Since we have found that Dutch has longer dependency distances on average, by going from English to Dutch we reverse the reordering rule with regard to the length of dependency lengths. Preferably we would have done it without moving to another language pair (direction), but since we want to conduct our experiments in natural languages, without an artificial long-distance-dependency-English as end-product, we need to do this. To test the effect of reducing head dependency distance without explicitly matching word order, we describe a reordering based on minimising head dependant distances, an idea we proposed in Zwarts and Dras (2006), and apply this to Dutch-English translation. We note that minimising of dependency distances is a general principle appearing in a number of guises in psycholinguistics, for example the work of Hawkins (1990). In this paper we exploit this idea to develop one general syntactically motivated reordering rule subsuming those of Collins et al. (2005).

The article is structured as follows: in section 2 we give some background on previous work of word reordering as preprocessing, on general word ordering in languages. To test the contribution of moving heads and dependants closer, in section 3 we describe an algorithm for minimising dependency distances and then discuss the experiment and results. The experiment to test the contribution of explicit word order matching and its results is described in section 4. The implications of these results are discussed in section 5 which leads to a conclusion in section 6.

2 Clause Restructuring

Collins et al. (2005) describe reordering based on a dependency parse of the source sentence. In their approach they have defined six hand-written rules for reordering German sentences. In brief, German sentences typically have the tensed verb in second position; infinitives, participles and separable verb particles occur at the end of the sentence. (Dutch, the language we will use, shares these same characteristics.) These six reordering rules are applied sequentially to the German sentence, which is their source language. Three of their rules reorder verbs in the German language, and one rule reorders verb particles. The other two rules reorder the subject and put the German word used in negation in a more English-like position. All their rules are designed to match English word ordering as much as possible. Their approach shows that adding knowledge about syntactic structure can significantly improve the performance of an existing state-of-the-art statistical machine translation system. We note that all of these rules also bring heads and dependants closer together, usually so that they are adjacent.

This idea seems particular well suited for translating from German to English, as intuition suggests that German has longer head-dependant distances than English. If this were the case, reordering of German, or of a similar language like Dutch with longer dependency distances, would thus move related words together, possible within the phrasal window of phrasal SMT, where their translations are already more likely to be together on the target (English) side, with its shorter dependency distances. To find empirical evidence of this, we calculated all the dependency distances over two languages in the Europarl corpus (Koehn, 2003).

Table 1 shows the percentage of words that are within a distance of n of their head. For $n = 1$ this is the percentage of words which are adjacent to their head, either directly to the left or to the right. These results are obtained on the same type of text (political extracts) and are based on parser results. (The effects of the parsers used are discussed later in this paper. For Dutch we have also done the counting for human annotated text, which showed little difference from the parser results.)

In the 5 and 10 word ranges, we find in Dutch roughly double the proportion of words which exceed this limit compared to English, increasing for even longer dependencies.

Regarding the work of Collins et al., we suggest two possible sources for the improvement obtained.

Match target language word order Although most decoders are capable of generating words in a different order than the source language, usually only simple models are used for this reordering. In Pharaoh (Koehn, 2004), for example, every word reordering between languages is penalised and only the language model can encourage a different order. If we can match the word order of the target language to a certain degree, we might expect an increase in translation quality, because we already have more explicitly used information of what the new word ordering should be.

Fitting phrase window The achievement of Phrase-Based SMT (PSMT) (Koehn et al., 2003) was to combine different words into one phrase and treat them as one unit. Yet PSMT only manages to do this if the words all fit together in the same phrase-window. If in a language a pair of words having a dependency relation are further apart, PSMT fails to pick this up: for example, verb particles in German which are distant from the governing verb. If we can identify these long distance dependencies and move these words together into the span of one phrase, PSMT can actually pick up on this and treat it as one unit. This means also that sentences not reordered can have a better translation, because the phrases present in that sentence might have been seen (more) before.

In the remainder of the paper, we will look at the contribution of each possible cause, using the language pair English and Dutch. To test the matching of word order without moving related words into the phrase window, we look at the reverse translation direction, English-Dutch. As there are no readily apparent transformations rules from Dutch to English that require heads and dependants to be moved

≤	1	2	3	4	5	10	15	20
English	56.6%	77.7%	87.0%	91.6%	94.1%	98.7%	99.7%	99.9%
Dutch	53.0%	74.9%	83.3%	88.1%	91.2%	97.3%	98.9%	99.5%

Table 1: Head to Child Dependency Distances

apart — and any case these would not parallel particularly closely the Collins-style rules — we instead just reverse the direction of the rules in this reversed translation direction as described in section 4. To test the effect of moving related words without explicitly matching word order, we propose the idea in the following section.

3 Minimal Dependency Reordering

In this section we describe the basis of the algorithm for reducing head-dependant distance without explicitly matching word order, then give details of the algorithm and the experimental setup for evaluating it.

3.1 Word Order Tendencies in Languages

In the field of psycholinguistics Hawkins (1990) argues that the word order in languages is based on certain rules, imposed by the human parsing mechanism. Therefore languages tend to favour some word orderings over others. He uses this to explain, for example, the universal occurrence of postposed sentential direct objects in Verb-Object-Subject (VOS) languages.

In his work, he argues that one of the main reasons for having certain word orders is that we as humans try to minimise certain distances between words, so that the sentence is easier to process. In particular the distance between a head and its dependants is important. An example of this is the English Heavy Noun Phrase shift. Take the following two sentence variants:

1. *I give* <NP> *back*
2. *I give back* <NP>

Whether sentence 1 or 2 is favourable, or even acceptable, depends on the size (heaviness) of the NP. If the NP is *it* only 1 is acceptable. When the NP is medium-sized, like *the book*, both are fine, but the longer the NP gets the more favourable 2 gets. When the NP reaches quite large sizes, (i.e. *the book that was given to me while* ...), native speakers will say 1 is no longer acceptable. Hawkins explains this by using head-dependant distances. In this example *give* is the head in the sentence; if the NP is short, both the NP and *back* are closely positioned to the head. The longer the NP gets the further away *back* is pushed. The theory is that languages tend to minimise the distance. The example illustrates this, by the length of the NP: if the NP gets too long, we prefer 2 over 1, because we want to have *back* close to its head *give*. We want to use this behaviour, preference of short dependency distances, in MT.

3.2 The Basic Algorithm

As in Collins et al. (2005), the reordering algorithm takes a dependency tree of the source sentence as input. For every node in this tree the linear distance, counted in tokens, between the node and its parent is stored. The distance for a node is defined as the closest distance to the head of that node or its children.

To illustrate the algorithm of this section we present the following two examples:

(1). Verb initial in VP:
normal order:
wij moeten dit probleem aanpakken
we must this problem tackle
reordered:
wij moeten aanpakken dit probleem
we must tackle this problem
reference:
we must tackle this particular problem

(2). Verb Particle reordering:
normal order:
het parlement neemt de resolutie aan
the parliament takes the resolution over
reordered:
het parlement neemt aan de resolutie
the parliament takes over the resolution
reference:
parliament adopted the resolution

As an example of the calculation of distances, the left tree of Figure 1 is the dependency tree for the normal order for example (1); nodes are annotated with the distance from the word to its governor.[1]

Based on the distance of the different child nodes we want to define an optimal reordering and pick that one. This means we have to score all the different reorderings. We want a scoring measure to do this that ignores the sign of distances so that they can be aggregated, and gives higher weight to longer dependency distances. Thus, similar to various statistical optimisation algorithms such as Least Mean Squares, we calculate the square of the Sum of the Head Distances (SHD^2) for each node n, defined as:

$$\text{SHD}^2(n) = \sum_{c \,\epsilon\, \textit{children}(n)} \textit{Distance}(c, n)^2$$

Alternatively an absolute value could have been selected,

[1]Note that in example 1 *probleem* gets a value of 1, although the word itself is 2 away from its head; we are measuring the distance from this complete constituent and not this particular word.

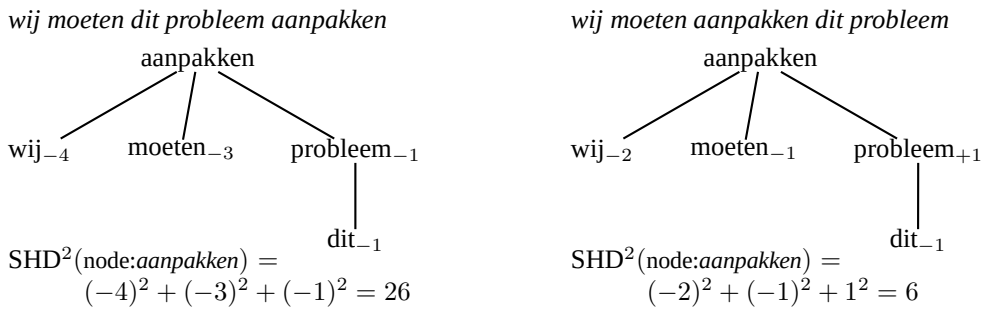

Figure 1: Reordering based on the sum of the head distances

but we want to penalise longer dependencies more heavily. A squared value delivers this property: a long distance dependency now results in a higher score than two short dependencies.

Every different ordering of children and head has a SHD^2 value; we are interested in minimising this value. We give an SHD^2 example in Figure 1.

We then reorder the children so that the SHD^2 score of a node is minimised. The right-hand tree of Figure 1 gives an example. In example 1 we can see how the Dutch verb *aanpakken* is moved to the beginning of the verb phrase. In this example we match English word order, even though this is not an explicit goal of the metric. The second example does not match English word order as such, but in Dutch the verb *aannemen* was split into *aan* and *neemt* in the sentence. Our reordering places these two words together so that PSMT can pick up that this is actually one single unit. Note that the two examples also demonstrate two of Collins et al. (2005) hand-written rules. In fact, this principle subsumes all the examples given in that paper in the prototypical cases.

In implementing the algorithm, for each node with children we calculate the SHD^2 for all permutations (note that this is not computationally expensive as each node has only a small number of children). We select the set of sibling orders with a minimal SHD^2. This is indeed a set because different orderings can still have the same SHD^2 value. To break tie we choose the member of the set with the fewest breaks in runs in the ordering of siblings. We describe this in more detail in Zwarts and Dras (2006).

To derive the dependency trees for Dutch, we used the Alpino (Bouma et al., 2000) parser.[2] Because this grammar comes with dependency information we closely follow their definition of head-dependant relations, deviating from this in only one respect. The Alpino parser marks auxiliary verbs as being the head of a complete sentence, while we took the main verb as the head of sentence, transforming the parse trees accordingly (thus moving closer to semantic dependencies).

The Alpino parser does not always produce projective parses. Reading off parse trees of Alpino in some cases already changes the word order, which leads to effects we discuss below.

[2]We would like to thank van Noord from the University of Groningen for kindly providing us the parses made by Alpino for most of the Dutch sections for the relevant data.

3.3 Reordering Models

We investigate four models for Dutch to English, with the aim of comparing an approach which minimises dependency distances with one based on Collins et al.

The first, the 'Alpino model', is to measure the impact of the parser used, as Alpino does some reordering that is intended to be linguistically helpful. We want to know what the result is of using a dependency parser which does not generate projective parses i.e. there are parses which do not result into the original sentence if we read off the tree for this parse. In this model we parse the sentences with our parser, and we simply read off the tree. If the tree is projective this results in the original tree. If this is not the case we keep for every node the original order of its children. We use this as our baseline, as the methods defined below use this word order as their starting point.

Our aim here is to define a method close to the Collins-style rules, which only reorder single-constituent units. First, however, we define our 'full model', which chooses the reordering as described in section 3.2. We expect the algorithm may be too 'enthusiastic' in reordering. For example, when we encounter a ditransitive verb the algorithm usually would put either the direct or the indirect object in front of the subject. Longer constituents were moved to completely different positions in the sentence. This kind of reordering could be problematic for languages, like English, which heavily rely on sentence position to mark the different grammatical functions of the constituents. More importantly we want to have a closer match to Collins et al. rules. In their experiment the vast majority of reordered chunks are single token constituents. For comparative reasons we keep the 'full model' and included that model in our experiments.

We therefore defined the 'limited model', a restriction on the previous model where only single tokens can be reordered. When analysing previous syntactically motivated reordering (Collins et al., 2005) we noticed that in most cases constituents consisting of one token only were repositioned in the sentence. Furthermore since sentence ordering is so important we decided only to reorder if 'substantial' parts were changed. To do this we introduced a threshold R and only accepted a new ordering if the new

SHD^2 has a reduction of at least R in regard to the original sentence ordering. If it is not possible to reduce SHD^2 that far we would keep the original ordering. We varied R between 0 and 1, for our final results in this work we used the value $R = 0.9$.

As a comparative we tried to reimplement the work of Collins et al. (2005). Given the availability of data and parsers, we tried to do this with the language pair Dutch-English, instead of the original German-English. Dutch and German share virtually all of the same grammatical rules for word order with respect to the Collins et al. rules.

We reimplemented the six rules in Collins et al. (2005) as closely as possible given our language pair and our parser, the Alpino parser. The 'Collins Model' will show us the impact of our parser and our choice of language pair. The changed rules, are implemented as follows:

1. **Verb Initial** In every constituent marked by Alpino as `VC` we move the head to first position. This results in moving verbs in a `VP` to the initial position.

2. **Verb 2nd** In subordinate clauses we move the verb to follow the complementiser. To recognise a subordinate clause we used constituents marked as `cp` by Alpino which have a `cmp` on its first position.

3. **Move Subject** In every clause we move the subject to immediately precede the head. Clauses are recognised by Alpino as `ssub` and `smain`.

4. **Particles** Verb particles are moved to immediately precede the verb. Verb particles are explicitly marked by Alpino.

5. **Infinitives** In the work of Collin et al., some verbs were not yet in the correct position. In this step they flatten the structure of the parse tree. Since in our work we already do this to make the right verb the head, this rule is disabled.

6. **Negation** This step tries to move the words indicating negation, like *not*, to the right position. Since these words are not explicitly marked by Alpino we search for specific lexical items. For all modifiers which match the Dutch word *niet* or *nooit*, we move it to directly follow the head verb.

3.4 Experimental Setup

For our experiments we used the decoder Pharaoh (Koehn, 2004). We use the default settings for Pharaoh, so that the parameters are not tweaked to optimise this language pair. Possible higher absolute scores might be acquired by doing this, but we aim to as closely as possible the work of Collins et al. (2005), where no minimum error rate training was carried out.

For the phrase extraction we used our own implementation of the algorithm which is described in the manual of Pharaoh. As a language model we used the SRILM (Stolcke, 2002) toolkit. We used a trigram model with interpolated Kneser-Ney discounting.

	BLEU
Dutch to English	
Not Reordered	0.207
Alpino	0.198
Collins	0.208
Full	0.196
Limited	0.203

Table 2: Automatic evaluation metrics for the dependency minimising models

For a baseline we used the Pharaoh translation made with a GIZA++ (Och and Ney, 2003) training on unchanged text, and the same phrase extractor was used for all of our models.

As an automated scoring metric we used the BLEU (Papineni et al., 2002) and the F-Measure (Turian et al., 2003) evaluation metrics.[3]

For our training data we used the Dutch and English portions of most of the Europarl Corpus. Because one section of the Europarl corpus was not available in a parsed form, this was left out. After sentence aligning the Dutch and the English part we divided the corpus into a training and a testing part. From the original available Dutch parses we selected every 200th sentence for testing, until we had 1500 sentences. We have a little over half a million sentences in our training section.

3.5 Results

For the four different models, results are given in Table 2. The first interesting result is the impact of the parser used. In the Europarl corpus 29% of the sentences have a different word order when just reading off the Alpino parse compared to the original word order. It turns out that our results for the Alpino model do not improve over a model where no reordering is performed.

In the original Collins et al. work, the improvement over the baseline was from 0.252 to 0.268 (BLEU) which was statistically significant. Here, the starting point for the Collins reordering is the read-off from the Alpino tree; the appropriate baseline for measuring the improvement made by the Collins reordering is thus the Alpino model, and the Collins model improvement is (a comparable) 0.01 BLEU point.

The Full Reordering model in fact does worse than the Alpino model. However, in our Limited Reordering model, our scores show a limited improvement in both BLEU and F-Measure above the Alpino model score.

In this model only half of the sentences (49%) are re-

[3] In this article, we used our own implementations of the BLEU and the F-Measure score available from `http://www.ics.mq.edu.au/~szwarts/Downloads.php`

		'Limited Reordering' success
1	B	the democratisation process is web of launched
	L	the democratisation process has been under way
	R	the process of democratisation has only just begun
2	B	the cap should be revised for farmers to support
	L	the cap should be revised to support farmers
	R	the cap must be reorganised to encourage farmers
3	B	but unfortunately i have to my old stand
	L	but unfortunately i must stand by my old position
	R	but unfortunately i shall have to adhere to my old point of view
4	B	how easy it is
	L	as simple as that
	R	it is as simple as that
5	B	it is creatures with an sentient beings
	L	there are creatures with an awareness
	R	they are sentient creatures
		'Limited Reordering' failure
6	B	today we can you with satisfaction a compromise proposal put
	L	today we can you and i am pleased submitting a compromise proposal
	R	i am pleased to see that we have today arrived at a compromise motion
7	B	this is a common policy
	L	is this we need a common policy
	R	a common policy is required in this area

Table 3: Examples of translation, B: Baseline, L: Our Limited Reordering model, R: Human Reference

ordered compared to the original source sentences. But as mentioned in section 2 not-reordered sentences can also be translated differently because we hope to have a better phrase table. When we compare sentence orders from this model against the sentence ordering from the direct read-off from the Alpino parser 46% of the sentences have a different order, so our method does much more than changing the 29% changed sentences of the Alpino read-off up to 49%.

Table 2 shows that reordering English as a preprocessing step for translating Dutch, results in up to half a BLEU percentage point. In this model we explicitly match target language, yet for most reorderings we increase dependency length. Now we observe half a BLEU percentage point improvement when we optimise for dependency length, an improvement weaker than the original, but still positive.

Since BLEU scores are not always indicative of translation quality (improvement) (Callison-Burch et al., 2006) in Table 3 we present some examples where we actually produce better translations than the baseline, and below that some examples where the baseline beats our model on translation quality. Example 3 takes advantage of a moved verb; the original Dutch sentence here ends with a verb indicating that the situation is unchanged. Example 2 also takes advantage of a moved final verb. In example 4, the baseline gets confused by the verb-final behaviour.

4 From Short to Long Dependencies

We investigate one model with translation direction English to Dutch where matching target language results in building longer dependencies. The model for English to Dutch is built to reflect as close as possible the rules in Collins et al. (2005), but in the opposite direction. In this case, the results should help decide whether final reordering improvements are predominantly caused by target language matching, or by the fact that we fit the (P)SMT model mechanisms better.

4.1 Algorithm and Experimental Setup

Th experimental setup is as in 3.4. For English we use the commercial Connexor parser (Tapanainen and Järvinen, 1997). Since our English reordering model is not based on head-dependant distances directly but hand crafted rules, it is less important that the notion of what constitutes a head is slightly different in Connexor from Alpino.

To match the rules in reverse we propose the following rule set for reordering:

1. If the auxiliary verb 'do' is present as a head of a `NEG` element (i.e. 'not'), these two nodes are fused into one node marked as new `NEG` node.
2. All verbs are moved to the end of the verb phrase, maintaining the original verb sequence.
3. Inflected verbs are moved directly after the subject.
4. In subordinate clauses the inflected verb is moved to the end of the verb phrase.
5. Nodes marked as `NEG` are moved to the position before main verb in the verb phrase, or right after the main verb is that verb is inflected.

	BLEU
English to Dutch	
Not Reordered	0.162
Reordered	0.167

Table 4: Automatic evaluation metrics for the matching target language models

Phrasal verbs are not yet moved, as most phrasal verbs in Dutch come from non-phrasal verbs in English.

4.2 Results

Again as seen in table 4 we observe half a BLEU point percent increase when we explicitly match target language word order, while not minimising dependency distances, but rather increasing them. We note that this half a BLEU point is from a different base (0.162 vs 0.198); proportionally the improvement is weaker than the original Collins-style rules.

5 Discussion

The Full Reordering model, without the limitation of moving only one token constituent and the R threshold, reorders most of the sentences: 90% of the Dutch sentences get reordered. As can be seen from Table 4, our scores drop even further than using only the Alpino model. Getting too close to the ideal of limiting dependency distance, we actually move large clauses around so much, for a language which depends on word order to mark grammatical function, that the sentences get scrambled and lose too much information. Manually judging the sentence we can find examples where the sentence locally improved in quality, but overall most translations are worse than the model without reordering.

In addition, the phrase table for the Fully Reordered model is much smaller than the phrase table for the non-reordered model. At first, we thought this was due to the new model generalising better. For example, we find the verb particle more often next to the governing verb than in other contexts. However a better explanation for this in light of the negative results for this model is based on the GIZA++ training. Eventually the phrases are derived from the output of a GIZA++ training which iteratively tries to build IBM model 4 (Brown et al., 1993) alignments on the sentence pairs. When the source sentences are extremely reordered (e.g. an object moved before the subject) the distortion model of model 4 makes it harder to link these words, so eventually we would extract fewer phrases.

Comparing the results of the Limited model to the reimplementation of the Collins rules in this work, we see that we have achieved a weaker improvement of half the magnitude, without using any language-specific rules. That the approach works by bringing related words closer, in a way that can be taken advantage of by the phrase mechanisms of SMT without explicitly matching the syntax of the target language, suggests that this is a source of the improvement obtained by the reordering approach of Collins et al.

Comparing the results for translation going in a translation direction where we decrease dependency distances (Dutch to English) to a translation in a direction where we increase these distances (English to Dutch) we notice that we also get an improvement of half a BLEU percentage point, which is weaker than the original. We therefore assume that both effects are important in explaining as to why word reordering works in (P)SMT. We hypothesise that both effects are weaker when they are executed individually without the other, but unlike the popular assumption it is not only target language matching which is important. More word reordering issues might be important in (P)SMT, but dependency distance minimisation as well as target language matching seems important in explaining which mechanism is important in explaining quality improvement in the (P)SMT paradigm.

6 Conclusion and Future Work

Previous work has demonstrated that reordering of the text in the source language can lead to an improvement in machine translation quality. We have questioned the reasons as to why improvement might occur. Previously, methods either tried to acquire appropriate rules for reordering by learning, or have used hand-coded rules that take account of specific differences between language pairs. In this work, we have explored how a claimed universal property of language — that there is a tendency to minimise the distance between a head and its dependants — can be adapted to automatically reorder constituents in the source language. This leads to an improvement in translation quality when the source language, Dutch, is one where this tendency is less present than in the target language English. An improvement is also observed in the opposite experiment, where we increase dependency lengths, but match target language word order. This is when the translation direction is reversed. Each leads to an improvement, but one that is weaker than the original. This suggests that what is contributing to the improvement obtained in the reordering is both the collapsing of elements into the phrasal window, and a better model for word reordering, since current word reordering mechanisms in (P)SMT are fairly weak. In demonstrating this, we have shown that an algorithm based on generic dependency minimisation can result in half the improvement of an approach using language specific rules. In future work, we first of all aim to find out why the process of parsing the Dutch causes the initial degrading results, which does not appear to have been the case for German. This leads to the more general question of when parsers are problematic for this kind of approach and how to know when this is the case.

References

Gosse Bouma, Gertjan van Noord, and Robert Malouf. 2000. Alpino: Wide Coverage Computational Analysis of Dutch. In *Computational Linguistics in the Netherlands (CLIN)*.

Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The Mathematic

of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the Role of Bleu in Machine Translation Research. In *11th Conference of the European Chapter of the Association for Computational Linguistics: EACL*, pages 249–256.

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 (ACL)*, pages 531–540, Ann Arbor, Michigan, June.

Marta R. Costa-Jussà and José A. R. Fonollosa. 2006. Statistical Machine Reordering. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 70–76.

J. A. Hawkins. 1990. A parsing theory of word order universals. *Linguistic Inquiry*, 21:223–261.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54, Morristown, NJ, USA. Association for Computational Linguistics.

Philipp Koehn. 2003. Europarl: A Multilingual Corpus for Evaluation of Machine Translation Philipp Koehn, Draft, Unpublished.

Philip Koehn. 2004. Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models, Philipp Koehn, AMTA.

Daniel Marcu, Wei Wang, Abdessamad Echihabi, , and Kevin Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. In *Proceedings of EMNLP*, pages 44–52.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

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

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 (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June. Association for Computational Linguistics.

A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings Intl. Conf. on Spoken Language Processing*, volume 2, pages 901–904.

Pasi Tapanainen and Timo Järvinen. 1997. A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing. Washington D.C.: Association for Computational Linguistics*, pages 64–71.

Joseph P. Turian, Luke Shen, and I. Dan Melamed. 2003. Evaluation of machine translation and its evaluation. In *Proceedings of MT Summit IX*.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese Syntactic Reordering for Statistical Machine Translation. In *Proceedings of EMNLP*, 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 (Coling)*, pages 508–514.

Simon Zwarts and Mark Dras. 2006. This Phrase-Based SMT System is Out of Order: Generalised Word Reordering in Machine Translation. In *Proceedings of the Australasian Language Technology Workshop*, pages 149–156.