Using Shallow Syntax Information to Improve Word Alignment and Reordering for SMT

Josep M. Crego
TALP Research Center
Universitat Politècnica de Catalunya
08034 Barcelona, Spain
jmcrego@gps.tsc.upc.edu

Nizar Habash
Center for Computational Learning Systems
Columbia University
New York, NY 10115, USA
habash@ccls.columbia.edu

Abstract

We describe two methods to improve SMT accuracy using shallow syntax information. First, we use chunks to refine the set of word alignments typically used as a starting point in SMT systems. Second, we extend an N-gram-based SMT system with chunk tags to better account for long-distance reorderings. Experiments are reported on an Arabic-English task showing significant improvements. A human error analysis indicates that long-distance reorderings are captured effectively.

1 Introduction

Much research has been done on using syntactic information in statistical machine translation (SMT). In this paper we use *chunks* (shallow syntax information) to improve an N-gram-based SMT system. We tackle both the alignment and reordering problems of a language pair with important differences in word order (Arabic-English). These differences lead to noisy word alignments, which lower the accuracy of the derived translation table. Additionally, word order differences, especially those spanning long distances and/or including multiple levels of reordering, are a challenge for SMT decoding.

Two improvements are presented here. First, we reduce the number of noisy alignments by using the idea that chunks, like raw words, have a translation correspondence in the source and target sentences. Hence, word links are constrained (i.e., noisy links are pruned) using chunk information. Second, we introduce rewrite rules which can handle both short/medium and long distance reorderings as well as different degrees of recursive application. We build our rules with two different linguistic annotations, (local) POS tags and (long-spanning) chunk tags. Despite employing an N-gram-based SMT system, the methods described here can also be applied to any phrase-based SMT system. Alignment and reordering are similarly used in both approaches.

In Section 2 we discuss previous related work. In Section 3, we discuss Arabic linguistic issues and motivate some of our decisions. In Section 4, we describe the N-gram based SMT system which we extend in this paper. Sections 5 and 6 detail the main contributions of this work. In Section 7, we carry out evaluation experiments reporting on the accuracy results and give details of a human evaluation error analysis.

2 Related Work

In the SMT community, it is widely accepted that there is a need for structural information to account for differences in word order between different language pairs. Structural information offers a greater potential to learn generalizations about relationships between languages than flat-structure models. The need for these '*mappings*' is specially relevant when handling language pairs with very different word order, such as Arabic-English or Chinese-English.

Many alternatives have been proposed on using syntactic information in SMT systems. They range from those aiming at harmonizing (monotonizing) the word order of the considered language pairs by means of a set of linguistically-motivated reordering patterns (Xia and McCord, 2004; Collins et al., 2005) to others considering translation a synchronous parsing process where reorderings introduced in the overall search are syntactically motivated (Galley et al., 2004; Quirk et al., 2005). The work presented here follows the word order harmonization strategy.

Proceedings of the Third Workshop on Statistical Machine Translation, pages 53–61,
Columbus, June 2008.

Collins et al. (2005) describe a technique for preprocessing German to look more like English syntactically. They used six transformations that are applied on German parsed text to reorder it before passing it on to a phrase-based system. They show a moderate statistically significant improvement. Our work differs from theirs crucially in that our preprocessing rules are learned automatically. Xia and McCord (2004) describe an approach for translation from French to English, where reordering rules are acquired automatically using source and target parses and word alignment. The reordering rules they use are in a context-free constituency representation with marked heads. The rules are mostly lexicalized. Xia and McCord (2004) use source and target parses to constrain word alignments used for rule extraction. Their results show that there is a positive effect on reordering when the decoder is run monotonically (i.e., without additional distortion-based reordering). The value of reordering is diminished if the decoder is run in a non-monotonic way.

Recently, Crego and Mariño (2007b) employ POS tags to automatically learn reorderings in training. They allow all possible learned reorderings to be used to create a lattice that is input to the decoder, which further improves translation accuracy. Similarly, Costa-jussà and Fonollosa (2006) use statistical word classes to generalize reorderings, which are learned/introduced in a translation process that transforms the source language into the target language word order. Zhang et al. (2007) describe a similar approach using unlexicalized context-free chunk tags (XPs) to learn reordering rules for Chinese-English SMT. Crego and Mariño (2007c) extend their previous work using syntax trees (dependency parsing) to learn reorderings on a Chinese-English task. Habash (2007) applies automatically-learned syntactic reordering rules (for Arabic-English SMT) to preprocess the input before passing it to a phrase-based SMT decoder.

As in (Zhang et al., 2007), (Costa-jussà and Fonollosa, 2006) and (Crego and Mariño, 2007b), we employ a word graph for a tight coupling between reordering and decoding. However, we differ on the language pair (Arabic-English) and the rules employed to learn reorderings. Rules are built using both POS tags and chunk tags in order to balance the higher generalization power of chunks with the higher accuracy of POS tags. Additionally, we introduce a method to use chunks for refining word alignments employed in the system.

3 Arabic Linguistic Issues

Arabic is a morpho-syntactically complex language with many differences from English. We describe here three prominent syntactic features of Arabic that are relevant to Arabic-English translation and that motivate some of our decisions in this work.

First, Arabic words are morphologically complex containing clitics whose translations are represented separately in English and sometimes in a different order. For instance, possessive pronominal enclitics are attached to the noun they modify in Arabic but their translation precedes the English translation of the noun: *kitAbu+hu*[1] '*book+his* → *his book*'. Other clitics include the definite article *Al+* '*the*', the conjunction *w+* '*and*' and the preposition *l+* '*of/for*', among others. We use the Penn Arabic Treebank tokenization scheme which splits three classes of clitics only. This scheme is compatible with the chunker we use (Diab et al., 2004).

Secondly, Arabic verb subjects may be: pro-dropped (verb conjugated), pre-verbal (SVO), or post-verbal (VSO). The VSO order is quite challenging in the context of translation to English. For small noun phrases (NP), small phrase pairs in a phrase table and some degree of distortion can easily move the verb to follow the NP. But this becomes much less likely with very long NPs that exceed the size of phrases in a phrase table.

Finally, Arabic adjectival modifiers typically follow their nouns (with a small exception of some superlative adjectives). For example, *rajul Tawiyl* (lit. man tall) translates as '*a tall man*'.

These three syntactic features of Arabic-English translation are not independent of each other. As we reorder the verb and the subject NP, we also have to reorder the NP's adjectival components. This brings new challenges to previous implementations of N-gram based SMT which had worked with language pairs that are more similar than Arabic and English, e.g., Spanish and English. Although Spanish is like Arabic in terms of its noun-adjective order; Spanish is similar to English in terms of its subject-verb order. Spanish morphology is more complex than English but not as complex as Arabic: Spanish is like Arabic in terms of being pro-drop but has a smaller

[1] All Arabic transliterations in this paper are provided in the Buckwalter transliteration scheme (Buckwalter, 2004).

number of clitics. We do not focus on morphology issues in this work. Table 1 illustrates these dimensions of variations. The more variations, the harder the translation.

	Morph.	Subj-Verb	Noun-Adj
AR	hard	VSO, SVO, pro-drop	N-A, A-N
ES	medium	SVO, pro-drop	N-A
EN	simple	SVO	A-N

Table 1: Arabic (AR), Spanish (ES) and English (EN) linguistic features.

4 N-gram-based SMT System

The **baseline** translation system described in this paper implements a log-linear combination of six models: a *translation model*, a *surface target language model*, a *target tag language model*, a *word bonus model*, a *source-to-target lexicon model*, and a *target-to-source lexicon model*. In contrast to standard phrase-based approaches, the translation model is expressed in **tuples**, bilingual translation units, and is estimated as an N-gram language model (Mariño et al., 2006).

4.1 Translation Units

Translation units (or tuples) are extracted after reordering source words following the **unfold** method for monotonizing word alignments (Crego et al., 2005). Figure 1 shows an example of tuple extraction with the original source-side word order resulting in one tuple (**regular**); and after reordering the source words resulting in three tuples (**unfold**).

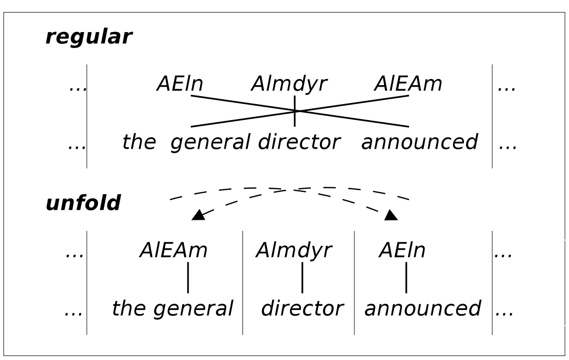

Figure 1: *Regular Vs. Unfold translation units.*

In general, the unfold extraction method outperforms the regular method because it produces smaller, less sparse and more reusable units, which is specially relevant for languages with very different word order. On the other hand, the unfold method needs the input source words to be reordered during decoding similarly to how source words were reordered in training. If monotonic decoding were used with unfolded units, translation hypotheses would follow the source language word order.

4.2 Reordering Framework

In training time, a set of reordering rules are automatically learned from word alignments. These rules are used in decoding time to provide the decoder with a set of reordering hypotheses in the form of a reordering input graph.

Rule Extraction

Following the **unfold** technique, source side reorderings are introduced into the training corpus in order to harmonize the word order of the source and target sentences. For each reordering produced in this step a record is taken in the form of a reordering rule: '$s_1, ..., s_n \rightarrow i_1, ..., i_n$', where '$s_1, ..., s_n$' is a sequence of of source words, and '$i_1, ..., i_n$' is a sequence of index positions into which the source words (left-hand side of the rule) are reordered. It is worth noticing that translation units and reordering rules are tightly coupled.

The reordering rules described so far can only handle reorderings of word sequences already seen in training. In order to improve the generalization power of these rules, linguistic classes (POS tags, chunks, syntax trees, *etc.*) can be used instead of raw words in the left-hand side of the rules. For example, the reordering introduced to unfold the alignments of the regular tuple '$AEln\ \ Almdyr\ \ AlEAm \rightarrow AlEAm\ \ Almdyr\ \ AEln$' in Figure 1 can produce the rule: '$VBD\quad NN\quad JJ \rightarrow 2\quad 1\quad 0$', where the left-hand side of the rule contains the sequence of POS tags ('*verb noun adjective*') belonging to the source words involved in reordering.

Search Graph Extension

In decoding, the input sentence is handled as a word graph. A monotonic search graph contains a single path, composed of arcs covering the input words in the original word order. To allow for reordering, the graph is extended with new arcs, covering the source words in the desired word order. For a given test sentence, any sequence of input tags fulfilling a left-hand side reordering rule leads to the

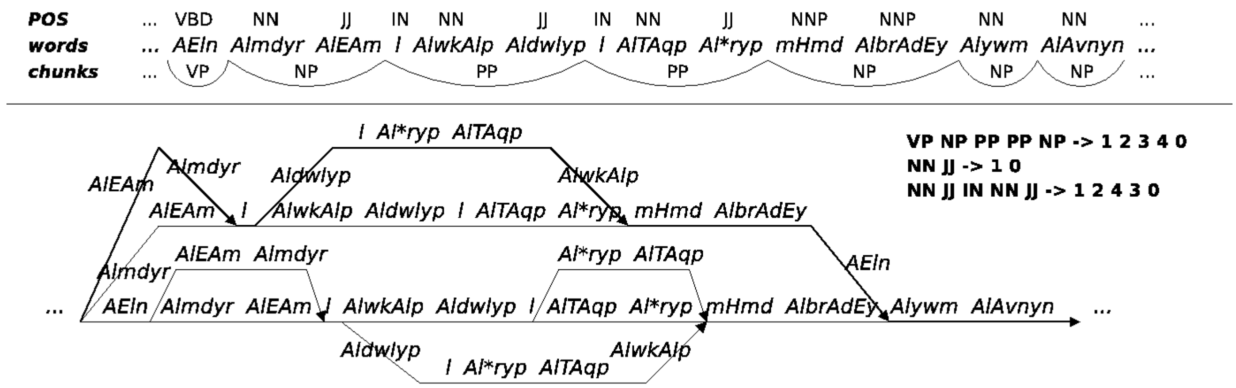

Figure 2: *Linguistic information, reordering graph and translation composition of an Arabic sentence.*

addition of a reordering path. Figure 2 shows an example of an input search graph extension (middle). The monotonic search graph is expanded following three different reordering rules.

5 Rules with Chunk Information

The generalization power of POS-based reordering rules is somehow limited to short rules (less sparse) which fail to capture many real examples. Longer rules are needed to model reorderings between full (linguistic) phrases, which are not restricted to any size. In order to capture such long-distance reorderings, we introduce rules with tags referring to arbitrary large sequences of words: chunk tags. Chunk-based rules allow the introduction of chunk tags in the left-hand side of the rule. For instance, the rule: '$VP \quad NP \rightarrow 1 \quad 0$' indicates that a verb phrase 'VP' preceding a noun phrase 'NP' are to be swapped. That is, the sequence of words composing the verb phrase are reordered at the end of the sequence of words composing the noun phrase.

In training, like POS-based rules, a record is taken in the form of a rule whenever a source reordering is introduced by the **unfold** technique. To account for chunk-based rules, a chunk tag is used instead of the corresponding POS tags when the words composing the phrase remain consecutive (not necessarily in the same order) after reordering. Notice that rules are built using POS tags as well as chunk tags. Since both approaches are based on the same reorderings introduced in training, both POS-based and chunk-based rules collect the same number of training rule instances.

Figure 3 illustrates the process of POS-based and chunk-based rule extraction. Here, the reordering

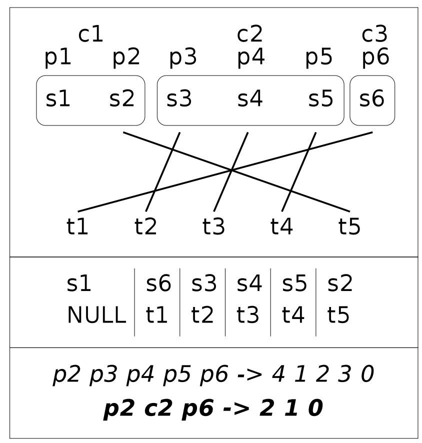

Figure 3: *POS-based and chunk-based Rule extraction: word-alignments, chunk and POS information (top), translation units (middle) and reordering rules (bottom) are shown.*

rule is applied over the sequence '$s_2 \ s_3 \ s_4 \ s_5 \ s_6$', which is transformed into '$s_6 \, s_3 \ \ s_4 \ \ s_5 \ \ s_2$'. As for the chunk rule, the POS tags '$p_3 \ p_4 \ p_5$' of the POS rule are replaced by the corresponding chunk tag 'c_2' since words within the phrase remain consecutive after being reordered. The vocabulary of chunk tags is typically smaller than that of POS tags. Hence, in order to increase the accuracy of the rules, we always use the POS tag instead of the chunk tag for single word chunks. In the example in Figure 3, the resulting chunk rule contains the POS tag 'p_6' instead of the corresponding chunk tag '$c3$'.

Any sequence of input POS/chunk tags fulfilling a left-hand side reordering rule entails the extension of the permutation graph with a new reordering path. Figure 2 shows the permutation graph (middle) computed for an Arabic sentence (top) after applying three reordering rules. The best path is drawn in bold arcs. It is important to notice that rules are *recursively* applied on top of sequences of already reordered words. Chunk rules are applied over phrases (sequences of words) which may need additional reorderings. Larger rules are applied before shorter ones in order to allow for an easy implementation of recursive reordering. Rules are allowed to match any path of the permutation graph consisting of a sequence of words in the original order. For example, the sequence '*Almdyr AlEAm*' is reordered into '*AlEAm Almdyr*' following the rule '$NN\ JJ \rightarrow 1\ 0$' on top of the monotonic path as well as on top of the path previously reordered by rule '$VP\ NP\ PP\ PP\ NP \rightarrow 1\ 2\ 3\ 4\ 0$'. In Figure 2, the best reordering path (bold arcs) could not be hypothesized without recursive reorderings.

6 Refinement of Word Alignments

As stated earlier, the Arabic-English language pair presents important word order disparities. These strong differences make word alignment a very difficult task, typically producing a large number of noisy (wrong) alignments. The N-gram-based SMT approach suffers highly from the presence of noisy alignments since translation units are extracted out of single alignment-based segmentations of training sentences. Noisy alignments lead to large translation units, which cause a loss of translation information and add to sparseness problems.

We propose an alignment refinement method to reduce the number of wrong alignments. The method employs two initial alignment sets: one with high precision, the other with high recall. We use the **Intersection** and **Union** (Och and Ney, 2000) of both alignment directions[2] as the high precision and high recall alignment sets, respectively. We will study the effect of various initial alignment sets (such as **grow-diag-final** instead of **Union**) in the future. The method is based on the fact that linguistic phrases (chunks), like raw words, have translation correspondences and can therefore be aligned. We use chunk information to reduce the number of allowed alignments for a given word. The simple idea that words in a source chunk are typically aligned to words in a single possible target chunk is used to discard alignments which link words from distant chunks. Since limiting alignments to one-to-one chunk links is perhaps too strict, we extend the number of allowed alignments by permitting words in a chunk to be aligned to words in a target range of words. This target range is computed as a projection of the source chunk under consideration. The resulting refined set contains all the Intersection alignments and some of the Union.

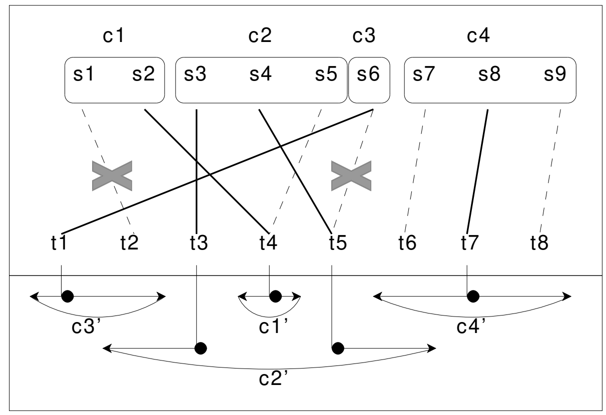

Figure 4: *Chunk projection: solid link are Intersection links and all links (solid and dashed) are Union links.*

We outline the algorithm next. The method can be decomposed in two steps. In the first step, using the Intersection set of alignments and source-side chunks, each chunk is projected into the target side. Figure 4 shows an example of word alignment refinement. The projection c'_k of the chunk c_k is composed of the sequence of consecutive target words $[t_{left}, t_{right}]$ which can be determined as follows:

- All target words t_j contained in Intersection links (s_i, t_j) with source word s_i within c_k are considered projection anchors. In the example in Figure 4, source words of chunk (c_2) are aligned into the target side by means of two Intersection alignments, (s_3, t_3) and (s_4, t_5), and producing two anchors (t_3 and t_5).

- For each source chunk c_k, t_{left}/t_{right} is set by extending its leftmost/rightmost anchor in the left/right direction up to the word before the next anchor (or the first/last word if at sentence edge). In the example in Figure 4, c'_1, c'_2, c'_3 and c'_4 are respectively $[t_4, t_4], [t_2, t_6], [t_1, t_2]$ and $[t_6, t_8]$.

In the second step, for every alignment of the Union set, the alignment is discarded if it links a

[2] We use IBM-1 to IBM-5 models (Brown et al., 1993) implemented with GIZA++ (Och and Ney, 2003).

source word s_i to a target word t_j that falls out of the projection of the chunk containing the source word. Notice that all the Intersection links are contained in the resulting refined set. In the example in Figure 4, the link (s_1, t_2) is discarded as t_2 falls out of the projection of chunk c_1 ($[t_4, t_4]$).

A further refinement can be done using the chunks of the target side. The same technique is applied by switching the role of source and target words/chunks in the algorithm described above and using the output of the basic source-based refinement (described above) as the high-recall alignment set, i.e., instead of Union.

7 Evaluation

7.1 Experimental Framework

All of the training data used here is available from the Linguistic Data Consortium (LDC).[3] We use an Arabic-English parallel corpus[4] consisting of 131K sentence pairs, with approximately 4.1M Arabic tokens and 4.4M English tokens. Word alignment is done with GIZA++ (Och and Ney, 2003). All evaluated systems use the same surface trigram language model, trained on approximately 340 million words of English newswire text from the English Gigaword corpus (LDC2003T05). Additionally, we use a 5-gram language model computed over the POS tagged English side of the training corpus. Language models are implemented using the SRILM toolkit (Stolcke, 2002).

For Arabic tokenization, we use the Arabic TreeBank tokenization scheme: 4-way normalized segments into conjunction, particle, word and pronominal clitic. For POS tagging, we use the collapsed tagset for PATB (24 tags). Tokenization and POS tagging are done using the publicly available Morphological Analysis and Disambiguation of Arabic (MADA) tool (Habash and Rambow, 2005). For chunking Arabic, we use the AMIRA (ASVMT) toolkit (Diab et al., 2004). English preprocessing simply included down-casing, separating punctuation from words and splitting off "'s". The English side is POS-tagged with TNT(Brants, 2000) and chunked with the freely available OpenNlp[5] tools.

We use the standard four-reference NIST MTEval data sets for the years 2003, 2004 and 2005 (henceforth MT03, MT04 and MT05, respectively) for testing and the 2002 data set for tuning.[6] BLEU-4 (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005) and multiple-reference Word Error Rate scores are reported. SMT decoding is done using MARIE,[7] a freely available N-gram-based decoder implementing a beam search strategy with distortion/reordering capabilities (Crego and Mariño, 2007a). Optimization is done with an in-house implementation of the SIMPLEX (Nelder and Mead, 1965) algorithm.

7.2 Results

In this section we assess the accuracy results of the techniques introduced in this paper for alignment refinement and word reordering.

Alignment Refinement Experiment

We contrast three systems built from different word alignments: (a.) the Union alignment set of both translation directions (U); (b.) the refined alignment set, detailed in Section 6, employing only source-side chunks (rS); (c.) the refined alignment set employing source as well as target-side chunks (rST). For this experiment, the system employs an n-gram bilingual translation model (TM) with $n = 3$ and $n = 4$. We also vary the use of a 5-gram target-tag language model (ttLM). The reordering graph is built using POS-based rules restricted to a maximum size of 6 tokens (POS tags in the left-hand side of the rule). The results are shown in Table 2.

Results from the refined alignment (rS) system clearly outperform the results from the alignment union (U) system. All measures agree in all test sets. Results further improve when we employ target-side chunks to refine the alignments (rST), although not statistically significantly. BLEU 95% confidence intervals for the best configuration (last row) are $\pm.0162$, $\pm.0210$ and $\pm.0135$ respectively for *MT03*, *MT04* and *MT05*.

As anticipated, the N-gram system suffers under high reordering needs when noisy alignments produce long (sparse) tuples. This can be seen by the increase in translation unit counts when refined links are used to alleviate the sparseness problem. The number of links of each alignment set over all

[3] http://www.ldc.upenn.edu

[4] The parallel text includes Arabic News (LDC2004T17), eTIRR (LDC2004E72), English translation of Arabic Treebank (LDC2005E46), and Ummah (LDC2004T18).

[5] http://opennlp.sourceforge.net/

[6] http://www.nist.gov/speech/tests/mt/

[7] http://gps-tsc.upc.es/veu/soft/soft/marie/

Align	TM	ttLM	BLEU	mWER	METEOR
MT03					
U	3	-	.4453	51.94	.6356
rS	3	-	.4586	50.67	.6401
rST	3	-	.4600	50.64	**.6416**
rST	4	-	.4610	50.20	.6401
rST	4	5	**.4689**	**49.36**	.6411
MT04					
U	3	-	.4244	50.12	.6055
rS	3	-	.4317	49.89	.6085
rST	3	-	**.4375**	49.69	**.6109**
rST	4	-	.4370	49.07	.6093
rST	4	5	.4366	**48.70**	.6092
MT05					
U	3	-	.4366	50.40	.6306
rS	3	-	.4447	49.77	.6353
rST	3	-	.4484	49.09	.6386
rST	4	-	.4521	48.69	.6377
rST	4	5	**.4561**	**48.07**	**.6401**

Table 2: *Evaluation results for experiments on translation units, alignment and modeling.*

training data is $5.5\,M$ (U), $4.9\,M$ (rS) and $4.6\,M$ (rST). Using the previous sets, the number of unique extracted translation units is $265.5\,K$ (U), $346.3\,K$ (rS) and $407.8\,K$ (rST). Extending the TM to order 4 and introducing the ttLM seems to further boost the accuracy results for all sets in terms of mWER and for MT03 and MT05 only in terms of BLEU.

Chunk Reordering Experiment

We compare POS-based reordering rules with chunk-based reordering rules under different maximum rule-size constraints. Results are obtained using TM $n = 4$, ttLM n=5 and rST refinement alignment. BLEU scores are shown in Table 3 for all test sets and rule sizes. Rule size $7R$ indicates that chunk rules are used with recursive reorderings.

BLEU	2	3	4	5	6	7	8	7R
MT03								
POS	.4364	.4581	.4656	**.4690**	.4689	.4686	.4685	-
Chunk	.4426	.4637	.4680	.4698	.4703	.4714	.4714	**.4725**
MT04								
POS	.4105	.4276	.4332	.4355	.4366	.4362	**.4368**	-
Chunk	.4125	.4316	.4358	**.4381**	.4373	.4372	.4373	.4364
MT05								
POS	.4206	.4465	.4532	.4549	.4561	.4562	**.4565**	-
Chunk	.4236	.4507	.4561	.4571	.4574	.4575	.4575	**.4579**

Table 3: *BLEU scores according to the maximum size of rules employed.*

Table 4 measures the impact of introducing reordering rules limited to a given size (Y axis) on the permutation graphs of input sentences from the MT03 data set (composed of 663 sentences containing $18,325$ words). Column *Total* shows the number of additional (extended) paths introduced into the test set permutation graph (*i.e.*, $2,971$ additional paths of size 3 POS tags were introduced). Columns 3 to 8 show the number of moves made in the 1-best translation output according to the size of the move in words (*i.e.*, $1,652$ moves of size 2 words appeared when considering POS rules of up to size 3 words). The rows in Table 4 correspond to the columns associated with MT03 in Table 3. Notice that a chunk tag may refer to multiple words, which explains, for instance, how 42 moves of size 4 appear using chunk rules of size 2. Overall, short-size reorderings are far more abundant than larger ones.

Size	Total	2	3	4	[5,6]	[7,8]	[9,14]
POS rules							
2	$8,142$	$2,129$	-	-	-	-	-
3	+$2,971$	$1,652$	707	-	-	-	-
4	+$1,628$	$1,563$	631	230	-	-	-
5	+964	$1,531$	615	210	82	-	-
6	+730	$1,510$	604	200	123	-	-
7	+427	$1,497$	600	191	121	24	-
8	+159	$1,497$	599	191	120	26	-
Chunk rules							
2	$9,201$	$2,036$	118	42	20	1	0
3	+$4,977$	$1,603$	651	71	42	5	2
4	+$1,855$	$1,542$	593	200	73	7	0
5	+$1,172$	$1,514$	578	187	118	15	1
6	+760	$1,495$	573	178	130	20	5
7	+393	$1,488$	568	173	129	27	10
8	+112	$1,488$	568	173	129	27	10
$7R$	+393	$1,405$	546	179	152	54	25

Table 4: *Reorderings hypothesized and employed in the* 1*-best translation output according to their size.*

Differences in BLEU (Table 3) are very small across the alternative configurations (POS/chunk). It seems that larger reorderings, size 7 to 14, (shown in Table 4) introduce very small accuracy variations when measured using BLEU. POS rules are able to account for most of the necessary moves (size 2 to 6). However, the presence of the larger moves when considering chunk-based rules (together with accuracy improvements) show that long-size reorderings can only be captured by chunk rules. The largest moves taken by the decoder using POS rules consist of 2 sequences of 8 words (Table 4, column 7, row 9 minus row 8). The increase in the number of long moves when considering recursive chunks ($7R$) means that longer chunk rules provide only valid reordering paths if further (recursive) reorderings are also considered. The corresponding BLEU score (Table 3, last column) indicates that the new set of moves improves the resulting accuracy. The general lower scores and inconsistent behavior of MT04 compared to MT03/MT05 may be a result of MT04 being a mix of genres (newswire, speeches and editorials).

7.3 Error Analysis

We conducted a human error analysis by comparing the best results from the POS system to those of the best chunk system. We used a sample of 155 sentences from MT03. In this sample, 25 sentences (16%) were actually different between the two analyzed systems. The differences were determined to involve 30 differing reorderings. In all of these cases, the chunk system made a move, but the POS system only moved (from source word order) in 60% of the cases. We manually judged the relative quality of the move (or lack thereof). We found that 47% of the time, chunk moves were superior to POS choice. In 27% of the time POS moves were better. In the rest of the time, the two systems were equally good or bad. The main challenge for chunk reordering seems to be the lack of syntactic constraints: in many cases of errors the chunk reordering did not go far enough or went too far, breaking up NPs or passing multiple NPs, respectively. Additional syntactic features to constrain the reordering model may be needed.

8 Conclusions and Future Work

In this work we have described two methods to improve SMT accuracy using shallow syntax information. First, alignment quality has been improved (in terms of translation accuracy) by pruning out noisy links which do not respect a chunk-to-chunk alignment correspondence. Second, rewrite rules built with two different linguistic annotations, (local) POS tags and (long-spanning) chunk tags, can handle both short/medium and long distance reorderings as well as different degrees of recursive application. In order to better assess the suitability of chunk rules we carried out a human error analysis which confirmed that long reorderings were effectively captured by chunk rules. However, the error analysis also revealed that additional syntactic features to constrain the reordering model may be needed. In the future, we plan to introduce weights into the permutations graph to more accurately drive the search process as well as extend the rules with full syntactic information (parse trees).

Acknowledgments

The first author has been partially funded by the Spanish Government under the AVIVAVOZ project (TEC2006-13694-C03) the Catalan Government under BE-2007 grant and the Universitat Politècnica de Catalunya under UPC-RECERCA grant. The second author was funded under the DARPA GALE program, contract HR0011-06-C-0023.

References

S. Banerjee and A. Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the Association for Computational Linguistics (ACL) Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

T. Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP'2000)*.

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

T. Buckwalter. 2004. Buckwalter Arabic Morphological Analyzer Version 2.0. Linguistic Data Consortium, University of Pennsylvania, 2002. LDC Cat alog No.: LDC2004L02, ISBN 1-58563-324-0.

M. Collins, P. Koehn, and I. Kucerova. 2005. Clause Restructuring for Statistical Machine Translation. In *Proceedings of ACL'05*.

M.R. Costa-jussà and J.A.R. Fonollosa. 2006. Statistical machine reordering. *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

J.M. Crego and J.B. Mariño. 2007a. Extending marie: an n-gram-based smt decoder. *Proceedings of ACL'07)*.

J.M. Crego and J.B. Mariño. 2007b. Improving statistical mt by coupling reordering and decoding. *Machine Translation*, 20(3):199–215.

J.M. Crego and J.B. Mariño. 2007c. Syntax-enhanced N-gram-based SMT. In *Proceedings of the Machine Translation Summit (MT SUMMIT XI)*.

J.M. Crego, J.B. Mariño, and A. de Gispert. 2005. Reordered search and tuple unfolding for ngram-based smt. *Proceedings of MT Summit X*.

M. Diab, K. Hacioglu, and D. Jurafsky. 2004. Automatic tagging of arabic text: From raw text to base phrase chunks. In *Proceedings of HLT-NAACL'04.*

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

N. Habash and O. Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of ACL'05.*

N. Habash. 2007. Syntactic Preprocessing for Statistical MT. In *Proceedings of MT SUMMIT XI.*

J.B. Mariño, R.E. Banchs, J.M. Crego, A. de Gispert, P. Lambert, J.A.R. Fonollosa, and M.R. Costa-jussà. 2006. N-gram based machine translation. *Computational Linguistics*, 32(4):527–549.

J.A. Nelder and R. Mead. 1965. A simplex method for function minimization. *The Computer Journal*, 7:308–313.

F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL.*

F. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–52.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL'02*

C. Quirk, A. Menezes, and C. Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of ACL'05*

A. Stolcke. 2002. SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP).*

F. Xia and M. McCord. 2004. Improving a statistical mt system with automatically learned rewrite patterns. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004).*

Y. Zhang, R. Zens, and H. Ney. 2007. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *Proceedings of HLT-NAACL Workshop on Syntax and Structure in Statistical Translation.*