Gappy Translation Units under Left-to-Right SMT Decoding

Josep M. Crego
Spoken Language Processing Group
LIMSI-CNRS, BP 133
91430 Orsay cedex, France
jmcrego@limsi.fr

François Yvon
Univ Paris-Sud 11
LIMSI-CNRS, BP 133
91430 Orsay cedex, France
yvon@limsi.fr

Abstract

This paper presents an extension for a bilingual n-gram statistical machine translation (SMT) system based on allowing translation units with gaps. Our gappy translation units can be seen as a first step towards introducing hierarchical units similar to those employed in hierarchical MT systems. Our goal is double. On the one hand we aim at capturing the benefits of the higher generalization power shown by hierarchical systems. On the other hand, we want to avoid the computational burden of decoding based on parsing techniques, which among other drawbacks, make difficult the introduction of the required target language model costs.

Our experiments show slight but consistent improvements for Chinese-to-English machine translation. Accuracy results are competitive with those achieved by a state-of-the-art phrase-based system.

1 Introduction

Work in SMT has evolved from the traditional word-based (Brown et al., 1993) to the current phrase-based (Och et al., 1999; Zens et al., 2002; Koehn et al., 2003) and hierarchical-based (Melamed, 2004; Chiang, 2007) translation models. Phrase-based and hierarchical systems are also characterized by the underlying formal device employed to produce translations (Knight, 2008): *finite-state transducers* (FST) on the one hand, and *tree transducers* (TT) on the other hand, specified respectively by rational and context-free grammars, thus implying clear differences in generative power.

A thorough comparison between phrase-based and hierarchical MT can be read in (Zollmann et al., 2008), concluding that hierarchical models slightly outperform phrase-based models under “sufficiently non-monotonic language pairs”. One of the reasons for the gap in performance seems to be the ability to generalize using non-terminal categories beyond the strictly lexicalized knowledge represented in phrase-based models.

An illustrative example is given below. It consist of the translation from English to French of negative verb phrases, which yields the alignment of *don’t X* ⇝ *ne X pas*, where X could be replaced by almost any finite verb. In this example, the English token *don’t* is translated into the French non-contiguous words *ne* and *pas* [1].

The right translation can only be achieved under phrase-based systems, if X (say *want*) has been seen in training next to *don’t*, yielding the translation unit:

$$don't\ want\ :\ ne\ veux\ pas$$

In contrast, under hierarchical systems, it is possible to obtain the right generalization, decomposing the previous pattern as:

$$X \rightarrow don't\ Y : ne\ Y\ pas$$
$$Y \rightarrow want : veux$$

[1] This example is only used for illustrative purposes. The contracted form *don’t* is not a real issue as most tokenizers split the form as *do not*, thus solving the alignment problem.

Proceedings of the 13th Annual Conference of the EAMT, pages 66–73,
Barcelona, May 2009

This ability to capture better generalization comes at a double price: translation as parsing is typically cubic with respect to the source sentence length; furthermore, in this formalism, target constituent are no longer produced monotonically from left-to-right, thus rendering the application of the language model score difficult (Chiang, 2007).

This example also suggests that hierarchical rules tend to be less sparse, given that the holistic unit in the phrase-based (PB) model is divided into two smaller, more reusable, rules. Notice that, in this specific case, the rich morphology of French verbs increases the sparseness problem of phrase-based translation units. Finally, by using discontinuous patterns, hierarchical translation models can capture large span (bilingual) dependencies.

Other than modeling discontinuous constituents, a major difference between FST- and CFG-based approaches to translation, has to do with the size of the search space, or more precisely with the kind of pruning that takes place to make the search feasible.

As previously outlined, when considering the use of translation units with gaps under the left-to-right decoding approach, the main difficulty arises motivated by the appearance of discontinuities in the output side. In this work, we make use of an input word lattice to naturally avoid this problem, allowing to monotonically compose translation.

Related Work

We follow the work in (Simard et al., 2005), which, to the best of our knowledge is the first MT system that within a left-to-right decoding approach, introduces the idea of phrases with gaps. A main limitation of their work arised from the difficulties of left-to-right decoders to handle gaps in the target side, again because of the non-monotonic generation of the target. Such gaps are to be filled in further steps of the search, thus, increasing the complexity of decoding and at the same time that hindering the use of the target language model.

Such translation units are more naturally used under systems employing parsing techniques to perform the search (hierarchical MT). Different kind of hierarchical translation units have been proposed, which mostly differ from the level of syntactical information they use. We mainly differentiate here between translation units that are formally syntax-based, like those appearing in (Chiang, 2007), which employ non-terminal categories without linguistic motivation, working as placeholders to be filled by words in further translation steps; and hierarchical units that are more linguistically motivated, as in (Zollmann and Venugopal, 2006).

More recently, (Watanabe et al., 2006) presents a hierarchical system in which the target sentence is generated in left-to-right order, thus enabling a straightforward integration of the n-gram language models during search. The authors employ a top-down strategy to parse the foreign language side, using a synchronous grammar having a GNF[2]-like structure. This means that the target side body of each translation rule takes the form $b\beta$, where b is a string of terminal symbols and β a (possibly empty) string of non-terminals. This ensures that the target is built monotonously. (Venugopal et al., 2007) present a hierarchical system that derives translations in two steps, so as to mitigate the computational impact resulting from the intersection of a probabilistic synchronous CFG and and the n-gram language model. Firstly, a CYK-style decoding considering first-best chart item approximations is used to generate an hypergraph of target language derivations. In the second step, a detailed exploration of the previous hypergraph is performed. The language model is used to drive the second step search process and to recover from search errors made during the first step.

Our work differs from theirs crucially in that our system employs a different set of translation structures (units), and because our decoder follows strictly the FST-based approach.

The remaining of this paper is organized as follows. In Section 2, we outline the n-gram-based approach used in the rest of this work. Sections 3 and 3.2 detail the use of translation units with gaps in a left-to-right decoding approach. Translation accuracy results are reported for the Chinese-English language pair in section 4. Finally, in section 5, we draw conclusions and outline further work.

[2] Greibach Normal Form

2 N-gram-based SMT

The baseline translation system described in this paper implements a log-linear combination of several models. In contrast to standard phrase-based approaches (Koehn et al., 2003), the translation model is expressed in *tuples* (instead of phrases), and is estimated as an N-gram language model over such units. It actually defines a joint probability between the language pairs under consideration (Mariño et al., 2006).

We have reimplemented the decoder described in (Crego and Mariño, 2007a), that we have extended to decode input lattices. At decoding time, only those reordering hypotheses encoded in the word lattice are to be examined. Reordering hypotheses are introduced following a set of reordering rules automatically learned from the bi-text corpus word-to-word alignments. Hence, reordering rules are applied on top of the source sentences to be translated.

More formally, given a source sentence, f, in the form of a linear word automaton, and N optional reordering rules to be applied on the given sentence in the form of string transducers (τ_i), the resulting lattice containing reordering hypotheses, f^*, is obtained by the sequential composition of FSTs, as:

$$f^* = \tau_N \circ \tau_{N-1} \cdots \circ \cdots \tau_1 \circ f$$

where $\circ$ denotes the composition operation.

Note that the sequence of FSTs (reordering rules) is sorted according to the length of the left-hand side (LHS) of the rule. More specific rules, having a larger LHS, are applied (composed) first, in order to ensure the recursive application of the rules. Hence, some paths are obtained by applying reordering on top of already reordered paths. Figure 1 illustrates an example where two reordering rules: $abc \leadsto cab$ (τ_1) and $ab \leadsto ba$ (τ_2) are applied on top of the sentence $abcd$ (s). As it can be seen, the resulting word lattice contains the path of the original sentence $s : abcd$, as well as the additional paths appeared by the composition of reordering rules: $\tau_1(s) : cab$, $\tau_2(s) : ba$ and $\tau_2(\tau_1(s)) : cba$.

Part-of-speech (POS) and syntactic information are used to increase the generalization power of our rules. Hence, instead of raw words, the LHS of the reordering rules typically make reference to POS-tags patterns, or to dependency sub-trees.

For instance, the rule $NN\ JJ \leadsto JJ\ NN$ is defined in terms of POS-tags, and produces the swap of the sequence *noun adjective* that is observed for the pair French-to-English. Additional details regarding the syntax-based rules are given in section 3.

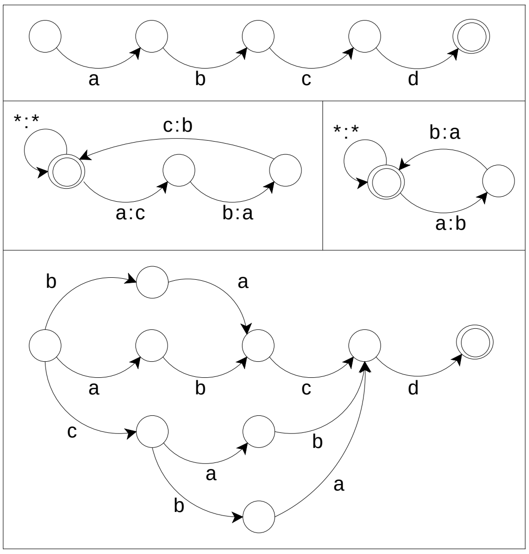

Figure 1: *Initial linear automaton (top). Reordering rules in the form of string transducers (middle) and final word lattice after rule composition.*

For the experiments reported in this paper, we consider that all paths in the input lattice are equally likely, a simplification we may wish to remove in further research.

3 Translation units with gaps

In this section we give details of the gappy translation units introduced in this work.

3.1 Split rules and reordering

Some phrase-based systems have been able to introduce some levels of syntactical information. In (Habash, 2007) the author employs automatically learned syntactic reordering rules to preprocess the input, aiming at solving the reordering problem, before passing the reordered input to a phrase-based decoder for Arabic-English translation. However, this kind of systems cannot produce the translation needed in our original English-to-French example because of the left-to-right decoding approach used in the underlying system. Translation is sequentially composed from left to right, and none of the word orderings of the source sentence, *don't + want* and *want + don't*, produces the desired translation. Instead, they produce respectively: *ne pas + veux* and *veux + ne pas*.

We propose a method that allows phrase-based systems to introduce gappy units similar to those typically employed in hierarchical systems, while keeping the left-to-right decoding approach.

To collect gappy units, we analyze the (symmetric) word alignments of the training corpus. The method basically consists of identifying, in the source sentence, single tokens translated into multiple ($n > 1$) non-contiguous target tokens. Figure 2 shows an example.

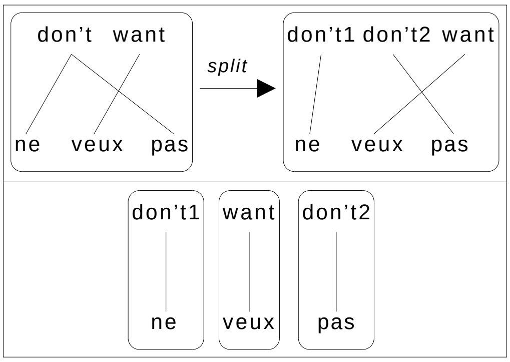

Figure 2: *Original tuple (top left), introduction of split words (top right) and tuples obtained after reordering source words (bottom).*

In the example, the English token $don't$ is translated into a sequence of discontinuous word segments *ne ... pas*. Once identified, the original source token is split so as to match the number of discontinuous segments. To continue with our example, $don't$ is split into $don't^1$ and $don't^2$ to match the two discontinuous segments *ne...pas*. Hence, similar to (Crego and Mariño, 2007b), we aim at monotonizing the word-to-word alignment, the main novelty being here the introduction of split tokens.

As it can be seen in the example, the target side of translation units remains unchanged, meaning that we can continue to generate the target in left-to-right fashion. Word reorderings and split words are introduced in the source sentence only, motivating the use of a word lattice. During training, the alignment is entirely monotonized before extracting tuples, only keeping those *one-to-many* and *many-to-one* alignments where the tokens on the *many* are contiguous; when this is not the case, splitting takes place.

Note that when translating the same example in the opposite direction, that is from French to English, the right translation is achieved without needing to split tokens. In such a case, the system would proceed by first reordering source words, obtaining *ne pas veux*, and then monotonically translating using the units: *ne pas : don't* and *veux : want*, yielding the right translation *don't want*.

When decoding test sentences, the word lattice is used to encode the most promising reorderings/splits of the input sentence, so as to reproduce the modifications introduced in the source sentences of the training corpus (as shown in figure 2). Thus, we slightly extend the reordering formalism introduced in 2 to allow the insertion of split tokens. Following the previous example, the new rule consists of:

$$don't \ want \ \rightsquigarrow \ don't^1 \ want \ don't^2$$

meaning that whenever you find in the input sentence the word sequence *don't want*, the input lattice is extended with the path $don't^1$ $want \ don't^2$, as represented on figure 3.

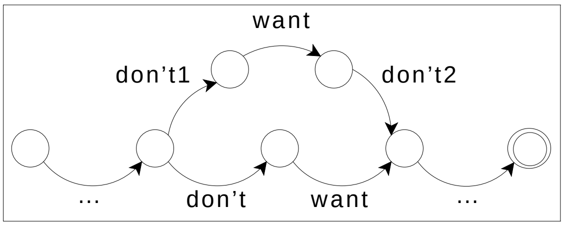

Figure 3: *Monotonic input graph extended with a split rule.*

So far, the method presented does not produce gappy units, but standard tuples with higher monotonization levels. However, with the addition of split rules, they become very similar to the units used in hierarchical translation systems. Note that the resulting extended input graph (figure 3) contains exactly the units extracted by the splitting procedure (figure 2 bottom).

The fully lexicalized split rules previously introduced would however be useless, failing to generalize to novel patterns. Therefore, as is done with "standard" reordering rules, split rules are defined over patterns of POS tags, instead words. Of course, the identity of split word has to be preserved, as it would make no sense to split, during decoding, words for which no translation units have been collected in training. Finally, the split rule induced for the previous example is:

$$don't\ V \ \leadsto\ \ don't^1\ V\ don't^2$$

where V is a POS tag standing for a verb.

This strategy has two additional benefits. First, it yields smaller translation units, whose probability are better estimated. Going back to the example of figure 2, the original translation unit (left) is larger than the new one (right), and more likely to cause estimation problems. Second, it allows to better use the information available in the training corpus. To see why, consider again our running example. Leaving the original unit undecomposed prevents to extract the match between *want* and *veux*, which is correctly extracted in the novel formalism.

In the next section, we detail how the generalization power of split/reordering rules can be further increased by using dependency parse trees.

3.2 Syntax aware split rules

Syntactic reordering rules employed in this work are similar to those detailed in (Crego and Mariño, 2007b). These rules introduce reorderings at the level of syntactic nodes. Hence, long reorderings can be achieved with short rules, as nodes may dominate arbitrary long sequences of words. Thus, the LHS of the rules is referred to the parse nodes of the original source sentences, while the RHS specifies the permutation that is introduced. Figure 4 shows the parse tree and POS tags of the Chinese sentence: *Aozhou shi yu Beihan you bangjiao de shaoshu guojia zhiyi*, an example borrowed from (Chiang, 2007).

Figure 5 illustrates how, by applying three rules to the previous Chinese example, we can get the reorderings/split required to derive the correct English translation: *Australia is one of the few countries that have diplomatic relations with North Korea.* As previously stated, rules (FSTs) are sorted before applied (composed). Note that in the case of syntactic rules, the length of a rule is based on the number of words appearing in the LHS of the rule.

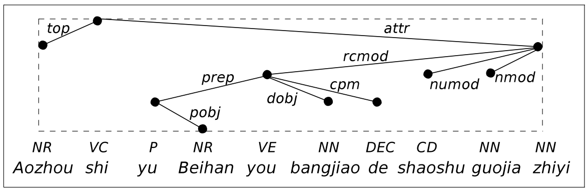
top
attr
rcmod
prep
cpm
dobj
numod
nmod
pobj
NR VC P NR VE NN DEC CD NN NN
Aozhou shi yu Beihan you bangjiao de shaoshu guojia zhiyi

Figure 4: *Dependency parse tree and POS tags of the Chinese sentence: 'Aozhou shi yu Beihan you bangjiao de shaoshu guojia zhiyi'.*

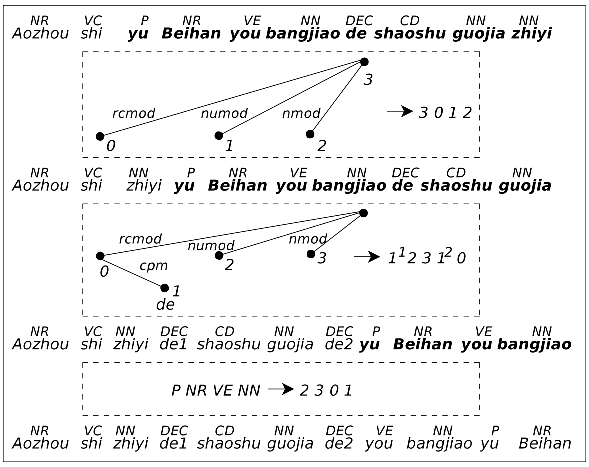
NR VC P NR VE NN DEC CD NN NN
Aozhou shi yu Beihan you bangjiao de shaoshu guojia zhiyi
rcmod
numod
nmod
0
1
2
3
→ 3 0 1 2
NR VC NN P NR VE NN DEC CD NN
Aozhou shi zhiyi yu Beihan you bangjiao de shaoshu guojia
rcmod
numod
nmod
cpm
0
1
de
2
3
→ 1¹2 3 1² 0
NR VC NN DEC CD NN DEC P NR VE NN
Aozhou shi zhiyi de1 shaoshu guojia de2 yu Beihan you bangjiao
P NR VE NN → 2 3 0 1
NR VC NN DEC CD NN DEC VE NN P NR
Aozhou shi zhiyi de1 shaoshu guojia de2 you bangjiao yu Beihan

Figure 5: *Chinese sentence rewritten by means of reordering/split rules.*

Considering the first rule applied in figure 5, the tree in its LHS contains four nodes (eight words), which cover the following sequences of Chinese words: *yu Beihan you bangjiao de*, *shaoshu*, *guojia* and *zhiyi*. Words matched by the rules are displayed above the rules using bold characters.

Note that equivalently to POS rules, words to be split in syntactical rules appear fully lexicalized. The second rule in figure 5 splits the word *de*. Thus, it appears fully lexicalized in the LHS of the rule.

Finally, the last rule is formed of POS tags. It reorders the words *yu Beihan you bangjiao* into *you bangjiao yu Beihan*. The monotonic translation of the resulting reordered path yields the correct English translation.

Syntactical reordering/split rules are automatically extracted from the training bi-texts, making use of the word-to-word alignments and the source dependency trees.

To conclude this section, notice that gappy units introduced in this work are only those that are motivated by word structures where words of the source side are aligned to multiple non-contiguous words of the target side. As a result, we approximate the behavior of a hierarchical system employing only a very limited set of rule patterns.

4 Experiments

In this section, we give details regarding the evaluation framework and report on the experimental work carried out to evaluate the improvements.

4.1 Evaluation Framework

We have used the BTEC (Takezawa et al., 2002) corpus focusing on translations from Chinese to English. It consists of the data made available for the IWSLT 2007 evaluation campaign. Some statistics regarding the corpora used, namely number of sentences, words, vocabulary, average sentence length and number of references per language are shown in table 1.

	Sent	Words	Voc	Avg	Refs
Train					
en	40k	377k	11k	9.5	1
zh		354k	9,6k	8.9	
Tune / Test (zh)					
tune	506	3,564	871	7	16
tst2	500	3,608	921	7.22	16
tst3	506	3,889	916	7.69	16
tst4	489	5,476	1,094	11.2	7
tst5	500	5,846	1,292	11.69	7
tst6	489	3,325	864	6.8	6

Table 1: *BTEC Corpus (Chinese-to-English).*

Chinese words were segmented by means of the ICTCLAS (Zhang et al., 2003) tagger/segmenter. Word alignments were computed for the training data in the original word order, using GIZA++[3]. The grow-final-diag-and heuristic is used to refine the alignments before the translation units extraction. The Chinese side was parsed using the freely available Stanford Chinese Dependency Parser[4]. We have used the SRILM toolkit[5] to estimate the N-gram language models, using respectively 4 and 5 as n-gram orders for the translation LM and target LM (Kneser-Ney smoothing and interpolation of lower and higher n-grams are always used).

For tuning, optimal log-linear coefficients were found using an in-house implementation of the downhill SIMPLEX method. The BLEU score was used as the objective function.

4.2 Results

Accuracy results are reported for different configurations in table 2. System configurations consist of: **base** for which translation units do not introduce the ability to split source words into multiple tokens, and **+split** where the previous technique is used. The **POS** configuration employs POS tags in the source side of the reordering rules while **+SYN** employs both POS tag and syntactic rules.

Set	base		+split		Moses
	POS	+SYN	POS	+SYN	
tst2	47.25	48.15	47.42	**48.39**	48.14
tst3	55.82	56.88	56.44	**57.17**	55.95
tst4	15.72	16.82	16.48	17.08	**18.06**
tst5	15.89	16.32	16.34	**16.89**	15.91
tst6	29.56	30.81	29.81	31.67	**31.76**

Table 2: Accuracy results measured using the BLEU score.

The last column shows accuracy results obtained by **Moses** (Koehn et al., 2007), a state-of-the-art phrase-based SMT system.

It is worth saying that the Moses system was built using the same data sets and alignments that were used for our system (Moses performs lexicalized reordering with a maximum reordering distance of 8 words). In this case, we run a different optimization for each of the system configurations. BLEU confidence intervals range depending on the test set approximately from ±2.0 to ±3.0 points BLEU.

As it can be seen, the system built using the **+split** technique obtains higher accuracy results than the baseline one (**base**), in all test

[3] www.fjoch.com/GIZA++

[4] nlp.stanford.edu/downloads/lex-parser.shtml

[5] www.speech.sri.com/projects/srilm

sets and for both reordering rule configurations (**POS** and **+SYN**).

Even if results show a clear tendency to highly score the **+split** system, differences in all BLEU results fall within the confidence margin. However, when inspecting translations obtained by the system **+split +SYN**, we find several examples, such as the one shown in figure 6, where the decoder succeeds to apply the proposed gappy units.

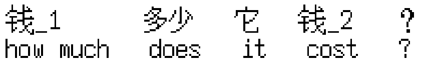

Figure 6: *Sequence of translation units output by the decoder.*

As it can be seen, motivated by a gappy unit, the first Chinese word is translated in two distant steps, yielding *how much* and *cost* respectively. The gap between both fragments is correctly filled by the English words *does it* as translation of the second and third Chinese words.

Considering the **base** systems, the same translation could only be produced if the first three Chinese words had been seen in training aligned to *how much does it.* In other words, larger units are needed to account for the correct translation.

The increment in the total number of translation units extracted when moving from the **base** to the **+split** configurations (from $267k$ to $285k$), as well as the increment in units used to translate the test sets (from $18,345$ to $19,150$) supports the fact that higher monotonizations levels of the training corpus have been achieved. All together, the resulting vocabulary of translation units, including all the new split units ($13,706$), contains $63,036$ units to be compared with the $56,046$ units in the baseline system.

Considering search efficiency, decoding time was increased about 1.5 times when building the system using the **split** technique, for both reordering rule configurations (**POS** and **+SYN**). Using gappy translation units does not increase the complexity of the search.

5 Conclusions and Further Work

In this paper, we have presented an extension to a bilingual n-gram translation system in which we allow translation units with gaps. The use of word lattices allowed us to introduce the concept of gappy translation units into an n-gram-based system, as an attempt to bridge the gap between phrase-based and hierarchical systems. Our decoder additionally benefits from the simplicity of left-to-right decoders, in contrast to the cost in complexity incurred by performing decoding as parsing. This have been achieved by means of standard tuples tightly coupled with reordering/split rules, introduced into the overall search through an input word lattice.

Our small but consistent accuracy improvements can mainly be attributed to the fact that a higher level of monotonization of the training corpus allows the extraction of smaller/more reusable units. As explained above, the split/reordering rules used in this study are costless, meaning that all reorderings are equally likely. As a consequence, the reward of using a split rule only comes from the translation models' score, which are computed separately for each instance of a split token. We believe that devising an appropriate weighting scheme for these split/reordering rules is needed to take full advantage of the extra expressiveness allowed by gappy units.

With the objective that our translation model highly benefits from the advantages of additional context, each gappy translation unit must be entirely weighted with a single probability. Instead, in our current implementation, each gappy unit is multiply weighted with partial probabilities. An open issue to definitely tackle in further research.

Additionally, we believe that the slight improvements achieved can be increased if additional gappy units are acquired from bilingual structures other than the one-to-many employed in the present experiments. We plan to extend the framework proposed in this paper with more complex gappy units, similiar to those used by hierarchical MT systems, thereby, taking full advantage of additional translation context provided by these units. We also plan to further investigate other aspects of hierarchical units, such as different levels of lexicalization in both the source and the target side.

Acknowledgments

This work has been partially funded by OSEO under the Quaero program.

References

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

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

Crego, J.M. and J.B. Mariño. 2007a. Extending marie: an n-gram-based smt decoder. *45rd Annual Meeting of the Association for Computational Linguistics*, April.

Crego, J.M. and J.B. Mariño. 2007b. Syntax-enhanced n-gram-based smt. *Proc. of the MT Summit XI*, pages 111–118, September.

Habash, N. 2007. Syntactic preprocessing for statistical machine translation. *Proc. of the MT Summit XI*, September.

Knight, Kevin. 2008. Capturing practical natural language transformations. *Machine Translation*, 21(2):121–133.

Koehn, Ph., F.J. Och, and D. Marcu. 2003. Statistical phrase-based translation. *Proc. of the Human Language Technology Conference, HLT-NAACL'2003*, May.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej 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, Prague, Czech Republic, June. Association for Computational Linguistics.

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

Melamed, D. 2004. Statistical machine translation by parsing. *42nd Annual Meeting of the Association for Computational Linguistics*, pages 653–661, July.

Och, F.J., Ch. Tillmann, and H. Ney. 1999. Improved alignment models for statistical machine translation. *Proc. of the Joint Conf. of Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, June.

Simard, M., N. Cancedda, B. Cavestro, M. Dymetman, E. Gaussier, C. Goutte, K. Yamada, P. Langlais, and A. Mauser. 2005. Translating with non-contiguous phrases. pages 755 – 762, October 6-8.

Takezawa, T., E. Sumita, F. Sugaya, H Yamamoto, and S. Yamamoto. 2002. Toward a broad-coverage bilingual curpus for speech translation of travel conversations in the real world. *3rd Int. Conf. on Language Resources and Evaluation, LREC'02*, pages 147–152, May.

Venugopal, Ashish, Andreas Zollmann, and Vogel Stephan. 2007. An efficient two-pass approach to synchronous-CFG driven statistical MT. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 500–507, Rochester, New York, April. Association for Computational Linguistics.

Watanabe, T., H. Tsukada, and H Isozaki. 2006. Left-to-right target generation for hierarchical phrase-based translation. *Proc. of the 21st Int. Conf. on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, July.

Zens, R., F.J. Och, and H. Ney. 2002. Phrase-based statistical machine translation. In Jarke, M., J. Koehler, and G. Lakemeyer, editors, *KI - 2002: Advances in artificial intelligence*, volume LNAI 2479, pages 18–32. Springer Verlag, September.

Zhang, H., H. Yu, D. Xiong, and Q. Liu. 2003. HHMM-based chinese lexical analyzer ictclas. In *Proc. of the 2nd SIGHAN Workshop on Chinese language processing*, pages 184–187, Sapporo, Japan.

Zollmann, Andreas and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 138–141, New York City, June. Association for Computational Linguistics.

Zollmann, Andreas, Ashish Venugopal, Franz Och, and Jay Ponte. 2008. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1145–1152, Manchester, UK, August. Coling 2008 Organizing Committee.