A Cocktail of Deep Syntactic Features for Hierarchical Machine Translation

Daniel Stein, Stephan Peitz, David Vilar, and Hermann Ney
Lehrstuhl für Informatik 6
RWTH Aachen University
Aachen, Germany
surname@informatik.rwth-aachen.de

Abstract

In this work we review and compare three additional syntactic enhancements for the hierarchical phrase-based translation model, which have been presented in the last few years. We compare their performance when applied separately and study whether the combination may yield additional improvements. Our findings show that the models are complementary, and their combination achieve an increase of 1% in BLEU and a reduction of nearly 2% in TER. The models presented in this work are made available as part of the Jane open source machine translation toolkit.

1 Introduction

Hierarchical phrase-based translation has proven to be one of the most successful approaches for statistical machine translation (Chiang, 2007). The approach can be considered to be a formal syntactic model, since the underlying structure is a grammar lacking linguistic knowledge. Given the increasing availability of linguistic parsers for many languages, hybrid approaches which incorporate deep syntactic knowledge often improve the translation quality. The goal is to enforce a more fluent grammar structure on the output hypotheses. Various groups report improvement over their baseline systems with different approaches, but it is not clear whether the benefits of the different methods are complementary or if they rather address the same issues.

In this work, we compare three recent syntactic methods that enhance the translation quality. We measure their performance individually and in combination with each other on a medium sized NIST Chinese-English task, and offer some analysis of typical translation examples. All the presented methods are released as part of the open source hierarchical machine translation toolkit Jane (Vilar et al., 2010).

This paper is organized as follows: We briefly recapitulate the hierarchical phrase-based model for machine translation in Section 2. We then describe the additional syntactical models used in this paper in Section 3. Results and detailed analysis on the NIST Chinese-English task are presented in Section 4. We conclude the paper in Section 5.

1.1 Related Work

One of the first papers to incorporate syntactic knowledge in a statistical machine translation model was (Yamada and Knight, 2001), although the performance was not on par with other state-of-the-art approaches at that time. Further development in this direction achieved competitive results, as can be seen in (DeNeefe et al., 2007) and later publications by the same group.

In contrast to these studies, which propose new models centered around the syntactic information, we focus mainly on methods that can be easily incorporated into an existing hierarchical system. In this work, we employ soft syntactic features comparable to (Vilar et al., 2008). These features measure how much a phrase corresponds to a valid syntactic structure of a given parse tree. Further, we include a dependency language model in a string-to-dependency model in the spirit of (Shen et al., 2008). We also derive soft syntactic labels as in (Venugopal et al., 2009), where the generic non-terminal of the hierarchical system is replaced by a syntactic label.

Other approaches in this field like (Chiang et al., 2009) and (Marton and Resnik, 2008) go into similar directions, but create a rather large quantity of features. We chose not to include their approaches into our comparison, since the risk of converging to poor local optima during the optimization procedure increases when too many features are available, thus making it difficult to draw clear conclusions.

2 Hierarchical Machine Translation

The hierarchical phrase-based approach can be considered to be an extension of the standard phrase-based model. In this model, we allow the phrases to have "gaps", i.e. we allow non-contiguous parts of the source sentence to be translated into possibly non-contiguous parts of the target sentence. The model can be formalized as a synchronous context-free grammar (Chiang, 2007). The bilingual rules are of the form

$$X \rightarrow \langle \gamma, \alpha, \sim \rangle , \tag{1}$$

where X is a non-terminal, γ and α are strings of terminals and non-terminals, and $\sim$ is a one-to-one correspondence between the non-terminals of α and γ.

Two examples of this kind of rules for the German-to-English translation direction are

$$X \rightarrow \langle \text{ich habe } X^{\sim 0} \text{ gesehen}, \text{I have seen } X^{\sim 0} \rangle$$
$$X \rightarrow \langle \text{um } X^{\sim 0} \text{ zu } X^{\sim 1}, \text{in order to } X^{\sim 1} X^{\sim 0} \rangle$$

where the indices in the non-terminals represent the correspondence between source and target "gaps". This model has the additional advantage that reordering is integrated as part of the model itself, as can be seen in the above examples.

2.1 Heuristics for Hierarchical Phrase Extraction

The phrase extraction process can be considered to consist of several steps, in each stage generating the phrases with a given number of non-terminals. The first step is the same as for the standard phrase-based model. Given a word aligned training corpus, phrases are typically derived by finding word groups in the source and target languages so that no alignment point is outside of this block for either language. The phrases thus extracted contain no gaps and are denoted as *initial phrases*.

Having this set of initial phrases, we search for phrases which contain other smaller sub-phrases and produce new phrases with gaps. This process is iterated until the phrases with the desired maximum number of gaps are extracted.

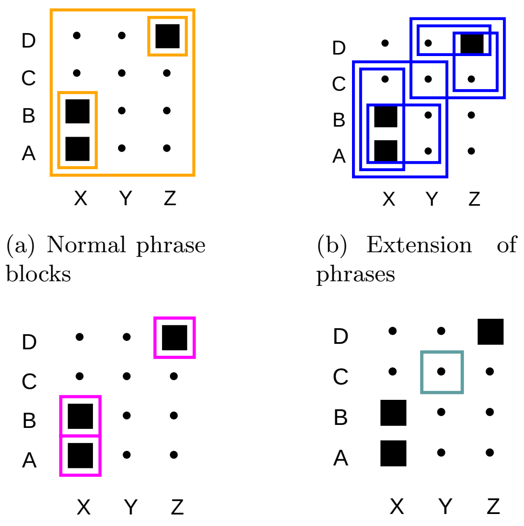
In our system, we apply common settings and restrict the number of non-terminals for each hierarchical phrase to a maximum of two, which are also not allowed to be adjacent on the source side. The gaps are allowed to span a maximum of 10 words. The phrase translation probabilities are computed as relative frequencies.

For language pairs like Chinese-English, the alignments naturally tend to be very non-monotonic. In addition, many words are left unaligned when using grow-diag or similar alignment combination heuristics. These artifacts may affect the quality of the extracted phrases (both initial and hierarchical) and degrade translation performance.

In order to counteract this effect we apply the following three heuristics for the extraction of initial phrases:

1. We expand the initial phrases to cover unaligned words adjacent to the phrase boundaries. Note that actually this is not a heuristic, but a strict application of the above (informal) definition of phrases. Common practice however is to extract only the shorter phrases that fulfill the given condition. The principle is illustrated in Figure 1(b).

2. We extract all single-word pairs derived from each alignment point, even if it violates the phrase rule mentioned above. For example, two consecutive words in one language aligned to a single word in the other are typically only extracted as a complete



(a) Normal phrase blocks

(b) Extension of phrases

(c) Single word heuristic

(d) Extraction of non-aligned word pairs

Figure 1: Extraction heuristics applied for initial phrases

phrase, but we also include each word pair as an independent phrase with a low probability. An example can be seen in Figure 1(c). The motivation is to be able to produce translations, possibly only partial, for all the words that have been seen in the training data, even if the word does not show up in the same context.

3. Similarly, all unaligned source words are paired with the unaligned target words and produce additional initial phrases, also with a low probability. An example is shown in Figure 1(d).

The additional phrases that are generated when applying these heuristics are not considered for the later extraction of hierarchical phrases. This is due to the large number of phrases that could be extracted when considering the whole set of initial phrases, which would pose efficiency problems for the translation process. In our experiments, the heuristic methods already increased the number of initial phrases roughly by a factor of 2.

2.2 Non-Syntactic Features

Our baseline system follows the usual log-linear model formulation (Och and Ney, 2002). We employ 14 features as given in the following list:

- Phrase translation probabilities in source-to-target and target-to-source directions.
- Single word lexicon models at phrase level, also in both directions.
- Word and phrase penalties.
- Source-to-target and target-to-source length ratio features for each phrase.
- One binary feature for hierarchical rules, one for the glue rule and one for phrases that do not perform reordering.
- Three binary features activated when a phrase has been seen more than one, three or five times (Mauser et al., 2006).

We are aware that this is already quite a large quantity of models for minimum error rate training. However, these features proved to be very effective for the Chinese-English translation direction and resulted in a strong baseline.

3 Syntactic Features

Unlike other work, like e.g. (Galley et al., 2004), we are not enforcing any syntactical integrity during the extraction process. Instead we produce additional information for each phrase. We mark those phrases that do not fit in the model with a binary feature. In this way we allow the corresponding scaling factors to decide whether the phrase can still be used during decoding. This also means that a scaling factor of zero allows the decoder to fall back to the baseline system during the minimum error rate training.

We parse the English target sentences with the Stanford parser[1], which is able to produce deep syntactic parses as well as dependency structures (de Marneffe and Manning, 2008).

In the following we will present the three syntactic models that we analyze in this work.

[1]http://nlp.stanford.edu/software/lex-parser.shtml

3.1 Parse Matching

The first model that we employ is also the simplest one. Given a monolingual sentence (be it in the source or the target language) and the associated parse tree, we will say that a lexical phrase extracted from this sentence is syntactically valid if it corresponds to the yield of one of the nodes in the syntax tree.

With this model, we hope that we can guide the decoder to prefer phrases that are syntactically sound rather than using arbitrary word combinations that spread over the boundaries of syntactic constructs. Two features are derived from this procedure. The first one measures the relative frequency with which a given phrase did not exactly match the yield of any node. This feature is straightforward to compute for the initial phrases. We extend this concept to hierarchical phrases by considering them as valid if the originating initial phrase was syntactically valid and every phrase which was suppressed in order to generate the gaps was also syntactically valid.

In the second feature, we soften up this rather hard decision. For example, we might want to penalize phrases that miss a valid node by one word only less than others that have a bigger mismatch with the parse tree. The second feature thus measures the relative distance to the next valid node, i.e. the average number of words that have to be added or deleted to match a syntactic node, divided by the phrase length. Hierarchical phrases are treated in a similar way as above.

This approach corresponds to the "binary" and "relative" soft syntactic features described in (Vilar et al., 2008). In Figure 2, an example is given. A phrase consisting of the words "A China insurance" would be considered a valid phrase because it matches the syntactic node "NP". In contrast, the phrase consisting of the words "China insurance" does not match any node, and would need to either add one word (e.g. "A") or delete a word like "China" to match a valid node. It thus has a relative distance of 0.5. Note that, from this example, it is also apparent that the parser output is poor for some sentences since it does not seem to recognize the verb "starts" properly.

3.2 Soft Syntactic Labels

Another possibility to extend the hierarchical model and include syntax information is to extend the set of non-terminals in the hierarchical model from the original set of generic symbols to a richer, more syntax-oriented set. With this, we hope to improve the syntactic structure of the output sentence. For example there may be rules which ensure that there is a verb in the translation of every source verb phrase.

However, augmenting the set of non-terminals also restricts the parsing space and thus we alter the set of possible translations. Furthermore, it can happen that no parse can be found for some input sentences. To address this issue, our extraction is extended in a similar way as in the work of (Venugopal et al., 2009). In this model, the original generic non-terminal X is not substituted, rather the new non-terminals are appended as additional information to the phrases and a new feature is computed based on them. In this way the original parsing and translation spaces are left unchanged. In contrast to the above work, where the authors expand the set of linguistic non-terminals to include a large set of new symbols, we restrict ourselves to the non-terminals that are to be found in the syntax tree.

Each initial phrase is marked with the non-terminal symbol of the closest matching node as described in the above subsection. When producing hierarchical rules, the gaps are labelled with the non-terminal symbols of the corresponding phrases instead of the original generic non-terminal X. It is important to point out that the syntax information is extracted from the target side only, but the substitution of the corresponding non-terminal symbol is carried out both on the source and the target sides (with the same non-terminal on both sides).

For every rule in the grammar we store information about the possible non- terminals that can be substituted in place of the generic non-terminal X, together with a probability for each combination of non-terminal symbols. More formally, let S be the set of possible syntax non-terminals. Given a rule r with n gaps, we de-

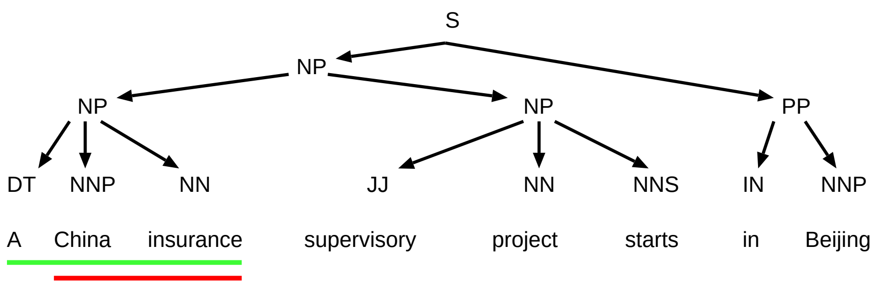


Figure 2: Deep syntactic parse for the sentence "A China insurance supervisory project starts in Beijing". For the parsematch feature, an example for a valid phrase is marked green and a non-valid one is marked red.

fine a probability distribution $p(s|r)$ over S^n+1, where s denotes a possible combination of syntax non-terminal symbols to be substituted in the rule, including the left-hand-side.

For each derivation d we compute two additional quantities. The first one be denoted by $p_h(Y|d)$ (h for "head") and reflect the probability that the derivation d under consideration of the additional non-terminal symbols has $Y \in S$ as its starting symbol. This quantity be needed for computing the probability $p_{\text{syn}}(d)$ that the derivation conforms with the extended set of non-terminals.

For the exact definition of these two quantities we separate the case where the top rule of derivation d is an initial phrase (in which case the derivation consists only of one rule application) and the general case where the top rule is a hierarchical one. If the top rule r of d corresponds to an initial phrase, the probability distribution for the non-terminals for d equals the distribution of rule r, i.e. $p_h(s|d) = p(s|r), \forall s \in S$. Given that only one rule has been applied, the derivation fully conforms with the extended set of non-terminals, thus in this case $p_{\text{syn}}(d) = 1$.

For the general case of hierarchical rules, let d be a general derivation, let r be the top rule and let $d_1, \ldots, d_n$ be the sub-derivations associated with the application of rule r in derivation d. For determining if the derivation is consistent with the extended set of non-terminals we have to consider every possible substitution of non-terminals in rule r and check the probability of the n sub-derivations to have the corresponding non-terminals. More formally:

$$p_{\text{syn}}(d) = \sum_{s \in S^{n+1}} \left(p(s|r) \cdot \prod_{k=2}^{n+1} p_h(s[k]|d_{k-1}) \right), \tag{2}$$

where the notation $[\cdot]$ denotes addressing the elements of a vector. The index shifting in the product in Equation 2 is due to the fact that the first element in the vector of non-terminal substitutions is the left-hand side of the rule, and this has to be taken into account when multiplying with the probabilities of the sub-derivations. Note also that although the sum is unrestricted, most of the summands will be left out due to a zero probability in the term $p(s|r)$.

The probability p_h is computed in a similar way, but the summation index is restricted only to those vectors of non-terminal substitutions where the left-hand side is the one for which we want to compute the probability. More formally:

$$p_h(Y|d) = \sum_{s \in S^{n+1}: s[1]=Y} \left(p(s|r) \cdot \prod_{k=2}^{n+1} p_h(s[k]|d_{k-1}) \right). \tag{3}$$

3.3 String-to-Dependency

Given a dependency tree of the target language, we are able to introduce language models that span over longer distances than shallow n-gram language models. In Figure 3, we can for example evaluate the left-handed dependency of the structure "In", followed by "industry", on the structure "faced". For this, we employ a

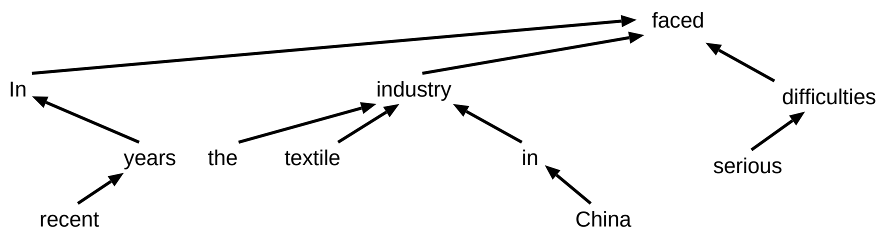


Figure 3: Dependency parsing for the sentence “In recent years, the textile industry in China faced serious difficulties”.

simple language model trained on dependency structures and compute the probability for the trigram “In industry faced-as-head”.

Rather than parsing the structures during decoding, we already apply the Stanford parser on the training material. Here, the sentences are generally well-formed and produce accurate parsing results, as opposed to an n-best list of the hypotheses. (Shen et al., 2008) use only phrases that meet certain restrictions. The first possibility is what the authors called a *fixed* dependency structure. With the exception of one word within this phrase, called the *head*, no outside word may have a dependency within this phrase. Also, all inner words may only depend on each other or on the head. For a second structure, called a *floating* dependency structure, the head dependency word may also exist outside the phrase. More formally:

Let dep_j denote the dependency of a word j on another word. A dependency structure $dep_{i...j}$ is called *fixed* on head h, iff

- $dep_h \notin [i, j]$
- $\forall k \in [i, j] \wedge k \neq h, dep_k \in [i, j]$
- $\forall k \notin [i, j], dep_k = h \vee dep_k \notin [i, j]$

and *floating* with children C for a non-empty set $C \subseteq \{i, \ldots, j\}$ iff

- $\exists h \notin [i, j],\ \text{s.t. } \forall k \in C, dep_k = h$
- $\forall k \in [i, j] \wedge k \notin C, dep_k \in [i, j]$
- $\forall k \notin [i, j], dep_k \notin [i, j]$.

We mark all phrases that meet these restrictions with a binary feature, but again do not limit the total phrase translation entry table. Additionaly, we store the dependency information in our phrase table, and further memorize for all hierarchical phrases if the gaps were dependent on the left or on the right side.

The approach in (Shen et al., 2008) relies on reconstructing the dependency tree of a hypothesis at decoding time by a bottom-up approach. In this work, we opted for a simpler approach and just penalize if the new phrase filling the gap in a larger hierarchical rule is pointing into the wrong direction, i.e. is pointing to the left when the hierarchical rule is expecting the gap to point to the right and vice versa. This introduces three features for merging errors to the left, merging errors to the right and the number of non-valid dependency structures used.

In a rescoring step on n-best lists, carried out after the decoding, we reconstruct as much of the dependency tree as possible. On these, we compute probabilities using three language models: one for left-side dependencies, one for right-side dependencies and one for head structures.

This approach has the drawback that badly reconstructed dependency trees tend to have only a few probability scores and thus to score higher than better structured trees in other sentences. We saw this effect in early experiments, even if we penalized every reconstruction error. Therefore, we decided to include a language count feature that is incremented each time we compute a dependency language model score, similar to the word penalty used for the normal language model.

		Chinese	English
Train	Sentences	3 030 696	
	# Words	77 456 152	81 002 954
	Vocabulary	83 128	213 076
	Singletons	21 059	95 544
Dev	Sentences	1 664	
	# Words	42 930	172 324
	Vocabulary	6 387	17 202
	OOVs	1 871	50 353
Test	Sentences	1 357	
	# Words	36 114	149 057
	Vocabulary	6 418	17 877
	OOVs	1 375	43 724

Table 1: Statistics for the Chinese-English corpus

4 Experimental Results

We used the Chinese-English NIST 2006 evaluation set as a development corpus and the NIST 2008 evaluation set as the blind test corpus. The systems were trained on a medium-sized training set. Statistics can be found in Table 1. All systems were optimized for the BLEU score using Och's MERT method (Och, 2003), with all scaling factors initialized with a value of 0.1. For rescoring with trigram dependency language models we generated 100-best lists after the optimization process.

Translation results obtained applying the methods discussed in Section 3 are shown in Table 2. All three methods yield improvements over the baseline system. The string-to-dependency method has very strong improvements in TER, while the soft syntactic labels perform very good in terms of BLEU. The parsematch approach is somewhat in between. The combination of the methods also leads to nice synergies. With the exception of the combination of parsematch and dependency, all pairs of combinations are better than their single feature systems. The combination of all three methods is best in terms of TER on the test set and only 0.3% worse than the best system on BLEU.

In Table 3, we present translation examples for two sentences, where the typical influence of the three methods on the translation outcome can be seen. The baseline translation usually produces most of the reference words and is understandable, but not very fluent. The parsematch approach prefers syntactically sound chunks but on the downside it often introduces unwanted fragments of a larger structure that do not make sense anymore in the given sentence. The dependency structure prefers sentences with a higher fluency but also seems to be more verbose, which is probably one of the main reasons why it fails to improve the n-gram precision score and thus the improvements in BLEU are less than in TER. The syntactic label features prefer similar labels as seen in the training. This generally works fine, but sometimes fails on sentences that are derived from newspaper headlines, as is often the case in the NIST corpus. In that case, an active form of the verb is usually preferred over a passive verb form, which leads to some decrease of the translation performance.

It seems that the methods all effect different aspects of the translation process, and that by combining these methods the resulting sentences are in general better than the baseline.

5 Conclusion

In this work, we considered several syntactically motivated enhancements for the hierarchical phrase-based translation system. We studied their influence on translation performance when applying them separately and how they interact with each other. The combination of the methods shows improvements over each single one. We can see that the models complement each other, as shown by the translation examples.

We also presented our implementation decisions, such as the heuristics applied at phrase extraction time. In addition, all these methods are released as open source for non-commercial purposes within the framework of Jane. We feel this is the right thing to do, as many times papers do not include enough details for replicating the results by other groups.

Acknowledgements

This work was partly realized as part of the Quaero Programme, funded by OSEO, French

	NIST '06 (dev)		NIST '08 (test)	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline hierarchical	31.37	63.16	24.04	68.37
parsematch	31.37	63.09	24.36	67.89
dependency	32.23	61.93	24.61	66.69
syntax labels	32.22	62.12	24.99	67.15
parsematch + dependency	32.03	62.45	24.61	67.57
syntax labels + parsematch	32.36	62.27	25.31	67.28
syntax labels + dependency	32.90	61.39	25.39	66.66
syntax labels + parsematch + dependency	32.89	60.99	25.11	66.39

Table 2: Results for the additional syntactic models on the NIST '06 and the NIST '08 test set

reference	Of course, I hope that all of these worries are needless.
baseline	Of course, I hope that all these are worried that is.
parsematch (par)	Of course, we hope that all concerned that this is groundless.
syntactic labels (syn)	Of course, I hope that all this concern is that.
dependency (dep)	Of course, we hope that all this worry is groundless.
dep + par	Of course, we hope that all these are worried about is the.
dep + syn	Of course, I hope that all this worry is unnecessary.
syn + par	Of course, I hope that all this worry is groundless.
dep + syn + par	Of course, I hope that all this worry is superfluous.

reference	I think I'm someone who doesn't easily lose his temper. That's my own feeling, at least.
baseline	I would like to the people I is not an easy miffed, at least , I feel this is the case.
parsematch	I think I am a person who is not easy miffed, at least this is my own feelings.
syntactic labels	I think I am a person who is not easy miffed, at least , I feel that the answer is in the affirmative.
dep	I think I am a person who is not easy miffed, at least this is my own feelings.
dep + par	I think I am a person who is not easy miffed, at least this is my own feelings.
dep + syn	I think I is a person who is not easy miffed, at least this is my own feelings.
syn + par	I think I am a person who is not easy miffed, at least this is my own feelings.
dep + syn + par	I think I am a person who is not easy miffed, at least i have the feeling that's the way it is.

Table 3: Example translations with the various methods

State agency for innovation and partly funded by the European Union under the FP7 project T4ME Net, Contract No. 249119.

References

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

David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228, June.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, Manchester, UK, August. Coling 2008 Organizing Committee.

Steve DeNeefe, Kevin Knight, Wei Wang, and Daniel Marcu. 2007. What can syntax-based MT learn from phrase-based MT? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 755–763.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule. *Proceedings of the Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Annual Meeting*, 4:273–280.

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1003–1011, Columbus, Ohio, USA, June.

Arne Mauser, Richard Zens, Evgeny Matusov, Saša Hasan, and Hermann Ney. 2006. The RWTH Statistical Machine Translation System for the IWSLT 2006 Evaluation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 103–110, Kyoto, Japan, November.

Franz Josef Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, PA, July.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, July.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A New String-to-Dependency Machine Translation Algorithm with a Target Dependency Language Model. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 577–585, Columbus, Ohio, June.

Ashish Venugopal, Andreas Zollmann, Noah A. Smith, and Stephan Vogel. 2009. Preference Grammars: Softening Syntactic Constraints to Improve Statistical Machine Translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 236–244, Boulder, Colorado, June.

David Vilar, Daniel Stein, and Hermann Ney. 2008. Analysing Soft Syntax Features and Heuristics for Hierarchical Phrase Based Machine Translation. In *International Workshop on Spoken Language Translation*, pages 190–197, Waikiki, Hawaii, October.

David Vilar, Daniel Stein, Matthias Huck, and Hermann Ney. 2010. Jane: Open source hierarchical translation, extended with reordering and lexicon models. In *ACL 2010 Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR*, pages 262–270, Uppsala, Sweden, July.

Kenji Yamada and Kevin Knight. 2001. A Syntax-based Statistical Translation Model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 523–530, July.