

Sub-Phrasal Matching and Structural Templates in Example-Based MT

Aaron B. Phillips

Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
aphillips@cmu.edu

Abstract

In this work I look at two different paradigms of Example-Based Machine Translation (EBMT). I combine the strengths of these two systems and build a new EBMT engine that combines sub-phrasal matching with structural templates. This synthesis results in higher translation quality and more graceful degradation, yielding 1.5% to 7.5% relative improvement in BLEU scores.

1 Introduction

Example-Based Machine Translation (EBMT) introduced the notion of phrasal translation that has subsequently been championed by Phrasal Statistical Machine Translation (PSMT). Exact phrasal translations are usually highly accurate and retain the nuances of the text. However, unless one focuses exclusively on a (very) small domain, it is unreasonable to assume that a corpus will provide exact phrasal translations of everything one wants to translate. Thus, methods of backing off and synthetically generating translations based on “similar” examples are increasingly important. In this work I introduce a new EBMT Engine named Cunei¹ (Construction of Unknown Examples by Induction) that combines two different paradigms of EBMT: sub-phrasal matching and structural templates. The goal of this work is to provide highly accurate translation when possible, but also allow

for more graceful degradation through a form of structural generalization.

2 Overview

The EBMT system at CMU, Panlite (Brown, 1996), is shallow in the sense that it only indexes lexical tokens. It performs well primarily because it is capable of indexing very large corpora and efficiently extracting exact lexical translations. When an example covering the full input sentence is not present in the corpus, Panlite attempts to match any sub-part of the sentence. This is done by matching all possible token sequences without any respect for phrasal boundaries. The retrieved examples are placed in a lattice that is subsequently decoded by a language modeler. This particular EBMT system is actually very similar to PSMT as it consists of a phrase extraction phase followed by a language modeler that performs phrase selection and reordering. The main differences lie in the details of the calculations and the fact that Panlite does not attempt to retain a true probabilistic model.

Not all EBMT implementations take this approach. In particular, Gaijin (Veale and Way, 1997) retrieves examples from a corpus based on their structural similarity. The marker hypothesis stipulates that a closed set of words in every language can be used to identify the syntactic structure of a sentence. These markers are typically conjunctions, prepositions, determiners, and quantifiers. Gaijin employs the marker hypothesis to segment sentences into constituent phrases as shown in Figure 1. Each constituent phrase is headed by a marker that represents the type of that constituent. The particular sequence of constituent phrases describes the structure of the sentence.

¹ Named after Cuneiform, the oldest writing system to be translated.



Figure 1. Sentence Segmented by Marker Hypothesis (Veale and Way, 1997)

This structure, rather than a language model, dictates phrasal selection and reordering. An example from the corpus that has the same sequence of constituent phrases becomes the master template for translation. When this template has lexical mismatches with the sentence to be translated, “grafting” is used to replace an entire phrasal constituent with another (more similar) phrasal constituent found in the corpus. Likewise, if particular words within a phrasal constituent do not match the input, “keyhole surgery” is performed to substitute individual lexical items. For either type of substitution to be performed, the structure (part-of-speech tag or head-of-phrase marker) must be equivalent.

Both of these EBMT systems build a final translation by synthetically combining together smaller units of translation. In the case of Panlite, the units are any sequence of lexical tokens, and they are combined together using a language modeler. On the other hand, the units in the Gaijin system are constituents identified by the marker hypothesis, and they are combined together by a single structural template from the corpus that matches the entire sentence.

Cunei attempts to bring together the strengths of Panlite and Gaijin. This new system maintains the indexing scheme and sub-phrasal matching found in Panlite and adds to this a “light” version of the structural matching found in the Gaijin system. Instead of using constituent phrases identified by the marker hypothesis as the structure of each sentence (Figure 1), Cunei uses only the sequence of part-of-speech tags as shown in Figure 2. Gaijin was built for a relatively small corpus and as such it was necessary to use a more general structure. The sequence of part-of-speech tags is very specific, but by leveraging a large corpus I expect to find many structural examples. This system will not, however, require one template to translate the entire sentence, but rather, like Panlite, will find

examples corresponding to any sub-section of the input sentence. Cunei passes the resulting lattice to the same language modeler used by Panlite for decoding.

Using part-of-speech tags to form structural templates is similar to the Transfer (Xfer) approach described in (Carbonell et al., 2002) and (Probst et al., 2003). The structural templates in Cunei are, in some respects, more limited as they do not incorporate morphological features. However, the role of the structural templates in Cunei is different as they are merely a backoff mechanism to be used when an exact lexical match is not present, and thus, generality is desired. In addition, the structural templates in Cunei are entirely data-driven. Instead of using a lexicon that specifies words available for substitution, Cunei fills the structural template using phrases present in the lattice that have the same part-of-speech sequence. The scores associated with each phrase in the lattice are taken into account when constructing a new example from the structural template.

Cunei was developed and evaluated translating text from Arabic to English. I expected the difference in word order between these two languages to work well with structural templates. However, the system is language-neutral and could easily be applied to any language pair for which part-of-speech taggers and parallel text are available.

3 Building Cunei

3.1 Preprocessing

For structural matching, it was important to process the English and Arabic in the same format as the Penn Treebank because this was expected by the part-of-speech taggers I used. A handful of regular expressions were applied to re-format the text and perform some simple cleanup. Next, I



Figure 2. A “Lite” Structure: Sentence with Part-Of-Speech Tags

used MXPOST (Ratnaparkhi, 1996) to apply part-of-speech tags to the English text and ASVMTools (Diab et al, 2004) to perform segmentation and part-of-speech tagging on the Arabic text. It is worthwhile to point out that because of the two different part-of-speech taggers, the naming conventions for the tags were not always the same. This does not make a difference to Cunei as there are no a priori rules that assume a noun should replace a noun. Rather, substitutions are determined at run-time based on the corpus and the alignment links.

3.2 Indexing

As mentioned previously, Cunei employs the same indexing approach used in Panlite, as this scales well with large amounts of data. The technique used in Panlite is to build a suffix array with the Burrows-Wheeler transform (Brown, 2004). Suffix arrays are an increasingly popular way to index large amounts of data and have been used as well by PSMT in (Zhang and Vogel, 2005) and (Callison-Burch, 2005). The Burrows-Wheeler transform brings the added benefit of considerably shrinking the size of the index.

In contrast to Panlite, Cunei needs to index the structure of the sentence as well as the lexical tokens. This was accomplished by using two indexes running in parallel as shown in Figure 3. Although this is not the most elegant approach, it is certainly the most practical approach. The two indexes allow for fast lookups of structural or lexical tokens. The downside is that the index is not optimized to look up combinations of structural and lexical tokens. To find the structural matches corresponding to a lexical match (or vice-versa), the sentence number and position within that sentence are identified and

looked up in the other index.

For lookups in the index, the Burrows-Wheeler transform does not result in any increase in computation. However, if one desires to reconstruct the text from the index, then looking up each type requires an additional binary search. For this reason, Cunei stores the index as a Burrows-Wheeler transformed suffix array on disk, but also allows for run-time reconstruction of the original suffix array. To reconstruct the original suffix array is very fast (linear transformation) but does require more memory. This is only performed when the task at hand requires reconstructing large amounts of the text and continuously looking up each type creates a performance bottleneck. For translation, it is usually necessary to reconstruct the suffix array for the target side of the index, but not the source side of the index.

Another optimization made in Cunei is to represent the index as a memory-mapped bit array. The bit array is dynamically adjusted to use the minimum number of bytes that are capable of representing the total number of types and tokens present in the corpus. This allows for a much smaller data structure than just representing everything with an integer, and (in theory) has no upper bound. Furthermore, the memory-mapped nature of the file makes the load time significantly faster. In this work I indexed 100,000 sentence pairs which only took a few minutes and consumed 27.5MB in all (including lexical and structural types and tokens for source and target).

3.3 Alignment

The second major component of the system is alignment. GIZA++ (Och and Ney, 2003) was used to generate a word alignment over the entire cor-

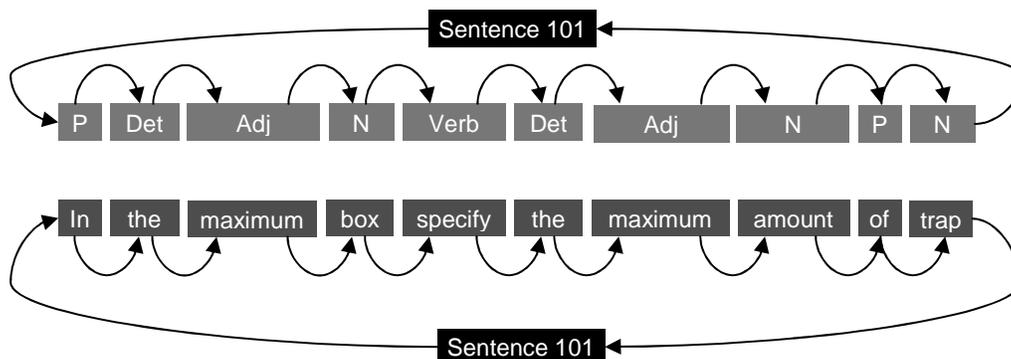


Figure 3. Indexing Structural and Lexical Tokens

pus. However, GIZA++ does not provide phrasal alignments which are necessary for translation. Thus, I investigated other alignment approaches and implemented a technique very similar to PESA (Vogel, 2005). The final alignment probability is calculated by taking the log-linear combination of the conditional probability of the entire source sentence given the target sentence, the conditional probability of the entire target sentence given the source sentence, and the length ratio between the selected source and target phrases. The conditional sentence probabilities are calculated by multiplying all the conditional word probabilities that agree with the phrasal alignment. A word alignment agrees with the phrasal alignment when it links two words that are both outside the phrasal alignment or two words that are inside the phrasal alignment.

3.4 Building Translations

Lexical translations are built by retrieving examples from the corpus and finding the aligned target text. Given a source text to translate, first Cunei looks in the source index for lexical examples of each sub-part of the source text. To ensure both speed and accuracy, a desired maximum number of instances of each distinct source phrase (typically 500-1000) is specified in a configuration file. If more than the desired number of examples are found, then the results are sub-sampled to only return the maximum. Each example is phrase aligned and the corresponding target text for each example is placed in a lattice. When more than one example produces the same target text, the results are merged together and their scores are combined.

This is the same basic approach used in Panlite and PSMT systems with online alignment such as those described in (Zhang and Vogel, 2005) and (Callison-Burch, 2005).

Where Cunei differs from other systems is that after all lexical look ups have been performed, Cunei looks for structural matches. Recall that the preprocessing routine has already tagged the source text with part-of-speech tags. Cunei queries the structural source index for all part-of-speech sequences that match a section of the input text's structure. A structural example is skipped if it is less than three tokens long or the maximum number of lexical examples has already been found for that section. In either of these cases, there is reason to believe that structural matches will not be useful. Similar to the lexical translations, once an example is found, it needs to be aligned to the target text. In this case the alignment extracts the target part-of-speech sequence rather than the lexical tokens. The retrieved part-of-speech sequence is used to predict the structure of the lexical target. This target part-of-speech sequence is converted to lexical example(s) through substitution. By following the alignment links, lexical translations present in the lattice are substituted into the structural template to form a new lexical translation. All elements in the lattice are searched to build lexical translations such that they maintain the same structure and alignment links as found in the structural example. An example of this is demonstrated in Figure 4. While single word substitutions are the most common, this process also looks for entire phrases that form an appropriate substitution. Furthermore, structural matches are analyzed from

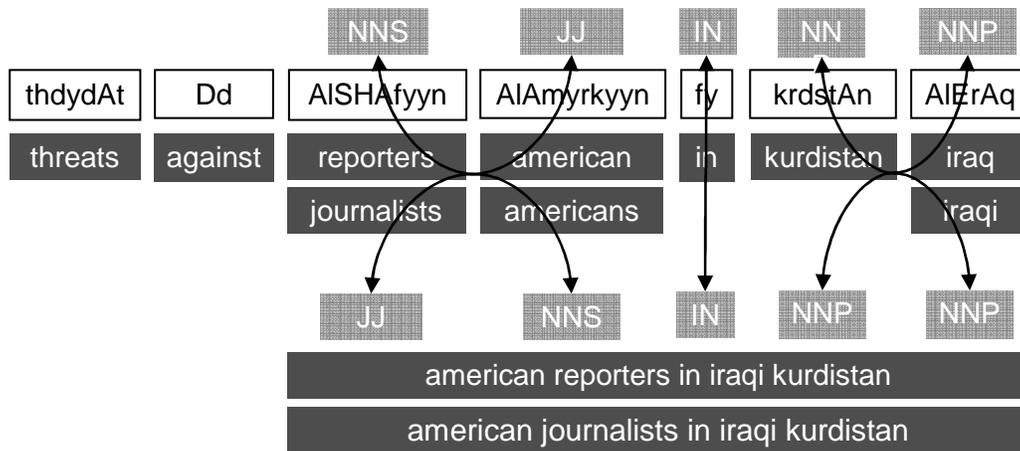


Figure 4. Example Constructed from Structural Template and Translation Lattice

shortest to longest so that longer matches can make use of translations created by shorter structural matches.

A minor exception to the process occurs when a structural example contains one or more lexical matches. To check for this situation, when a structural example is found, the lexical tokens of the structural example must be compared to the input text. When some of the lexical source tokens are the same, all target positions that align to a lexical source token are marked. These specially marked target positions cannot be replaced by other elements in the lattice. Rather, the lexical target tokens for these positions are retrieved from the corpus and used in the translation. This allows for structural examples where one or more source and target words are lexicalized even though the index does not directly support searching for this possibility.

3.5 Scoring Translations

Once all of the translations have been retrieved from the corpus or synthetically created from structural examples, it is necessary to score them. The language modeler will make the final decision as to which translations to use, but the language modeler must be provided with a score reflective of how likely each translation is to be representative of the source span it covers. In Cunei, each example that is placed in the lattice keeps track of three sub-scores: alignment probability, relative frequency, and context matches (the number of other examples in the lattice from the same sentence)². When two translations are merged because they share the same target translation, their sub-scores are added together. A final score is produced by a log-linear combination of the three sub-scores which are averaged over all found translations. The weights of the log-linear combination are defined in a configuration file and are tuned using held-out data.

The synthetic lexical examples built by combining long structural examples and shorter lexical examples pose a problem for scoring. As this specific lexical translation never occurs in the corpus, it is difficult to determine its relative frequency—a critical component of the scoring. Furthermore, the

distribution of the structural examples and the lexical examples is not the same, making the two relative frequencies hard to combine. Lastly, not all the structural examples are relevant to a particular input. Some structural examples can only occur with specific lexical elements or under specific conditions. Sometimes structural examples are found that cannot produce a lexical translation because the lattice lacks the necessary lexical items that match its structure and alignment constraints. Calculating relative frequency based on all the retrieved structural examples results in very low scores for each example, and it did not seem reasonable as many of these examples cannot occur for the given input.

To account for these differences, the relative frequencies for lexical and structural examples are calculated by only totaling over examples that produced a lexical translation. If the alignment process fails or if a structural example cannot find any appropriate lexical entries to create a lexical translation, then it is not included in the total count. In addition, a confidence score is applied to all translation candidates. If the translation candidate is retrieved from the corpus, then its confidence is 1.0. If the translation candidate is formed by a structural example, then its confidence score is the geometric mean of the scores of each lexical translation that was used (through substitution) to create the translation. This confidence score is an approximate measure of how closely a structural example matches the original source text. The confidence score is applied as a weight to each score when two translations are merged. Thus, an example with a low confidence score will not affect the overall scores as much as an example with a high confidence score. In practice this means that if a structural example predicts one target and a lexical example predicts a different target, the lexical example's target will have a higher score.

4 Results

Cunei was trained on approximately 100,000 sentence pairs (4.87 million words) of Arabic-English newswire text. This represents all available Arabic-English newswire text from the Linguistic Data Consortium with sentences containing fewer than 50 words. While more parallel Arabic-English data is available, most of it is out of domain and in the form of United Nations proceedings. The training

² Fully implemented in the system, but due to a significant slowdown in speed and very minor improvement in translation quality, the context score was disabled for the final results.

	MT03 (Tune)		News A		News B		Editorial		Speech		Full MT04	
Lexical and Reorder	0.444		0.483		0.455		0.321		0.339		0.397	
Structural and Reorder	0.452	1.65%	0.490	1.52%	0.475	4.38%	0.329	2.58%	0.364	7.52%	0.412	3.75%
Lexical no Reorder	0.419	-5.80%	0.461	-4.45%	0.434	-4.64%	0.320	-0.31%	0.333	-1.59%	0.385	-3.01%
Structural no Reorder	0.446	0.44%	0.490	1.51%	0.470	3.18%	0.333	3.83%	0.363	7.03%	0.411	3.57%

Figure 5. Table of Evaluation Results

data has good lexical coverage and at the same time is not prohibitively large for the structural matching.

Parameters for Cunei and the language modeler were tuned using part of the 2003 NIST MT Evaluation data set (MT03). However, due to time restraints, parameters for Cunei (as opposed to the language modeler) were not separately tuned for the system with structural matching enabled. Rather, I used the same parameters that were tuned on the system with structural matching disabled. Thus, these results do not reflect the full potential of the system with structural matching enabled.

Evaluation was performed by comparing Cunei with structural matching disabled to Cunei with structural matching enabled. This experiment was run twice: first with language model reordering enabled, and second with it disabled (monotonic decoding). All systems were evaluated on the 2004 NIST MT Evaluation data set (MT04), which provides five reference translations. MT04 contains editorial, speech, and news genres, but nearly half of it is news. I split MT04 by genre but also divided the news genre into two parts—one from Xinhua News Agency and the other from Agence France Press. Document boundaries were preserved in all the splits and the chunks range in size from 278 sentences to 387 sentences. Splitting the

data in this fashion allowed multiple evaluations on different types of data while maintaining enough sentences to have meaningful results. In addition, a final score for all of MT04 is provided.

The results are shown in Figure 5 and Figure 6. It is clear that the structural matching improves translation quality as BLEU scores improved under all testing conditions. While the relative improvement is smallest for “News A”, this is still a respectable gain in performance considering the high baseline. “News B”, “Editorial”, and “Speech”, which all have lower baselines, show stronger gains from the structural matching. This correlates well to the initial hypothesis that structural matching will make the system more robust and allow it to degrade more gracefully.

As expected, when language model reordering is disabled, the performance of the system with only lexical matching drops. This is not true for the system with structural matching enabled—signifying that the structural matching is capturing most (if not all) of the reordering.

Figure 7 and Figure 8 illustrate visually the differences in the types of translations found between the lexical only system and the structural system.

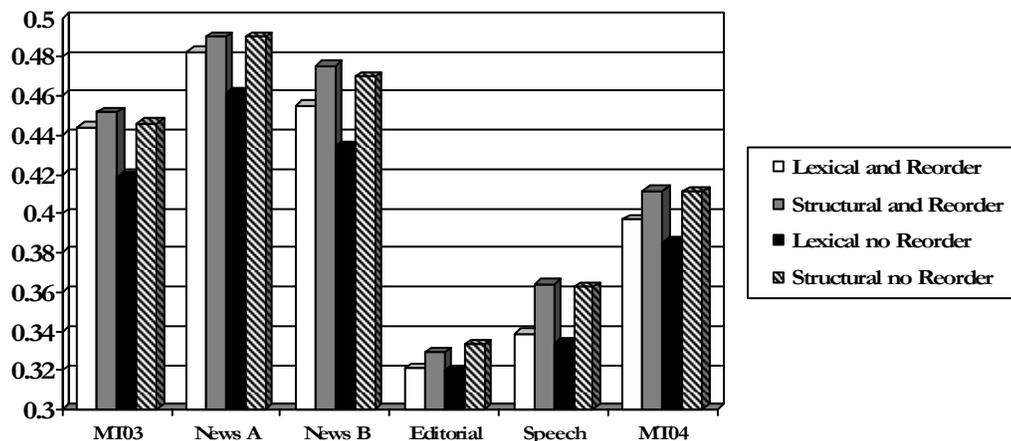


Figure 6. Chart of Evaluation Results

AlqrwD	AlkAfyp	tdEm	Alnmw	AlAqtSAdy	AlSyny
loans	enough	supports	growth	economic	chinese
loan	sufficient	support	development	the economic	the chinese
debts	sufficient guaranties	supported	the growth	iktisadi	china
the loans		supporting	growth rate	iktisadi	of chinese
of loans		supported by	of growth	of economic	of china
				chinese economic	
				the chinese economic	
				economic growth	
				the economic growth	
				economic progress	
				economic development	
				economic growth for	
				chinese economic growth	
				chinese economic development	
				chinese economic progress	
				support economic growth	
				support iktisadi growth	
				support iktisadi growth	

Figure 7. Translation Lattice with Structural Matching

AlqrwD	AlkAfyp	tdEm	Alnmw	AlAqtSAdy	AlSyny
loans	enough	supports	growth	economic	chinese
loan	sufficient	support	development	the economic	the chinese
debts	sufficient guaranties	supported	the growth	iktisadi	china
the loans		supporting	growth rate	iktisadi	of chinese
of loans		supported by	of growth	of economic	of china
				chinese economic	
				the chinese economic	
				economic growth	
				the economic growth	
				economic progress	
				economic development	
				economic growth for	

Figure 8. Translation Lattice without Structural Matching

5 Remaining Issues and Future Work

The problem of combining scores from two different probability distributions is fundamentally hard and the solution is not readily apparent. Applying confidence weights seemed reasonable, but I imagine much better solutions exist. Even if the confidence weights were retained, it would be worthwhile to investigate applying them in a non-linear fashion. Time limitations prevented experimentation with other methods.

Figure 7 and Figure 8 illustrate another problem: phrases inserted into the lattice do not always have optimal boundaries. The last three words “Alnmw AlAqtSAdy AlSyny” form one noun phrase that translates as “chinese economic growth”. The lexical system only provides “economic growth” and “chinese economic”. The structural matching does create “chinese economic growth”, but it also has partial translations of “economic growth”, “chinese economic”, and “support economic growth”. The problem is that these partial translations sometimes inappropriately guide the language modeler. Both the lexical and structural systems are affected by this issue, but the problem occurs with greater frequency when structural matches are enabled. This problem brings up the question of what makes a suitable translation unit. I did experiment with restrictions similar to those in the Gaijin system by limiting which part-of-speech tags a phrase is allowed to begin and end with. However, all of these experiments that “fil-

tered” the lattice resulted in lower scores. It would be worthwhile to investigate how to select more appropriate translation units, but in the meantime it appears to do more good than harm to allow all possible phrases.

Perhaps the most apparent “problem” with forming lexical translations from structural examples is speed. Enabling structural matching significantly slows down the system. It is for this reason that I did not tune all the parameters of the structural engine. The problem is that there are usually a lot of structural examples found in the corpus, and there are also a multitude of lexical translations that can be substituted into each structural example. The issue with speed is not due to poorly written code, but to the thousands of combinations that need to be analyzed for a match per example. The longer the example is, the more prone it is to this problem. I have partially alleviated this problem by pruning and chunking the input into smaller units. However, this merely makes the computation tractable, and not fast. More aggressive pruning and/or heavy caching techniques truly should be investigated.

6 Conclusion

In conclusion, this research describes a system that synthesizes two different approaches to EBMT. Whereas the origins of this system lie with EBMT, the end result is hard to classify as an EBMT system. Cunei has borrowed heavily from ideas and techniques present in EBMT, PSMT, and Xfer.

What is clear from this work, however, is that a data-driven approach that combines exact lexical matching with structural templates improves translation quality.

Acknowledgements

I would like to thank Ralf Brown for the reviewing this paper and his insightful thoughts throughout the research.

References

- Ralf D. Brown. 1996. Example-Based Machine Translation in the Pangloss System. In *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, pp. 169--174.
- Ralf D. Brown. 2004. A Modified Burrows-Wheeler Transform for Highly-Scalable Example-Based Translation. In *Machine Translation: From Real Users to Research, Proceedings of the 6th Conference of the Association for Machine Translation*, Washington, DC, pp. 27--36.
- Chris Callison-Burch, Colin Bannard and Josh Schroeder. 2005. Scaling Phrase-Based Statistical Machine Translation to Larger Corpora and Longer Phrases. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, Ann Arbor, Michigan, pp. 255--262.
- Jaime Carbonell, Katharina Probst, Erik Peterson, Christian Monson, Alon Lavie, Ralf Brown, Lori Levin. 2002. Automatic Rule Learning for Resource Limited MT. In *Proceedings of 5th Conference of the Association for Machine Translation in the Americas*, Tiburon, California, pp. 1--10.
- Mona Diab, Kadri Hacioglu and Daniel Jurafsky. 2004. Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. In *Proceedings of HLT-NAACL 2004*.
- Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1): pp. 19--51.
- Katharina Probst and Lori Levin and Erik Peterson and Alon Lavie and Jaime Carbonell. 2003. MT for Minority Languages Using Elicitation-Based Learning of Syntactic Transfer Rules. *Machine Translation*, 17(4): pp. 245--270.
- Adwait Ratnaparkhi. 1996. A Maximum Entropy Model for Part-of-Speech Tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 133--142.

Tony Veale and Andy Way. 1997. Gaijin: A Template-Driven Bootstrapping Approach to Example-Based Machine Translation. In *Proceedings of NeMNLP97, New Methods in Natural Language Processing*, Sofia, Bulgaria.

Stephan Vogel. 2005. PESA: Phrase Pair Extraction as Sentence Splitting. In *Proceedings: The Tenth Machine Translation Summit*, Phuket, Thailand.

Ying Zhang and Stephan Vogel. 2005. An Efficient Phrase-to-Phrase Alignment Model for Arbitrarily Long Phrase and Large Corpora. In *Proceedings of the Tenth Conference of the European Association for Machine Translation*, Budapest, Hungary.