Combining Semantic and Syntactic Generalization in Example-Based Machine Translation

Sarah Ebling*, **Andy Way****, **Martin Volk***, **Sudip Kumar Naskar****
* Institute of Computational Linguistics, University of Zurich, Binzmühlestrasse 14, 8050 Zurich, Switzerland
{ebling,volk}@ifi.uzh.ch
** CNGL, School of Computing, Dublin City University, Dublin 9, Ireland
{away,snaskar}@computing.dcu.ie

Abstract

In this paper, we report our experiments in combining two EBMT systems that rely on generalized templates, *Marclator* and *CMU-EBMT*, on an English–German translation task. Our goal was to see whether a statistically significant improvement could be achieved over the individual performances of these two systems. We observed that this was not the case. However, our system consistently outperformed a lexical EBMT baseline system.

1 Introduction

The state-of-the-art approach in MT is phrase-based Statistical Machine Translation (SMT) (Koehn et al., 2003). Together with Example-Based Machine Translation (EBMT) (Nagao, 1984), SMT belongs to the Corpus-Based Machine Translation (CBMT) paradigm. Hence, both SMT and EBMT rely on a sententially aligned bilingual corpus. EBMT systems make use of the parallel corpus by consulting the training set (their example base) directly at runtime. In contrast, SMT systems consult the probabilities of source-language–target-language (SL–TL) word or phrase pairs which they have learned from the training data offline. Hence, the main feature that distinguishes the two paradigms is the type of knowledge used during the translation step.

EBMT systems have often performed worse than SMT systems in the past (cf., for example, Groves and Way (2005)). The biggest shortcoming of EBMT is that it does not combine translations of phrases well. This problem is known as *boundary friction* (Way, 2001, p. 2). It is particularly frequent when translating into a morphologically rich language. As an example for translating from English into German, assume that the sentence pairs listed in Example 1 are contained in the example base (Way, 2001).

(1) ***A big dog*** *eats a lot of meat.* – ***Ein großer Hund*** *frisst viel Fleisch.*
I have *two ears.* – ***Ich habe*** *zwei Ohren.*

An EBMT system might make use of the phrases shown in bold to translate a sentence like *I have a big dog.* into *Ich habe ein großer Hund.* In doing so, it would neglect the fact that German uses different inflectional forms to mark grammatical case: the German phrase *ein großer Hund* in the first sentence is a nominative noun phrase and therefore a legitimate choice as the subject of this sentence, but *Ich habe* requires an accusative object (*einen großen Hund*).

Among the best-performing systems in EBMT are systems that make use of *generalized templates*. Generalized templates are SL–TL pairs in which certain parts have been replaced by variables. They provide an additional layer of abstraction and can thus prevent a system from having to revert to word-by-word translation.[1] In this paper, we present our experiments in combining two existing EBMT systems that rely on generalized templates. Our goal was to see whether a statistically significant improvement over the individual performances of these two systems could be achieved. We will show that this was not the case, but that our system performed significantly better than a lexical EBMT baseline system.

[1]It is generally accepted that translating a sentence word by word leads to poorer translation quality than translating it in larger segments.

Mikel L. Forcada, Heidi Depraetere, Vincent Vandeghinste (eds.)
Proceedings of the 15th Conference of the European Association for Machine Translation, p. 209–216
Leuven, Belgium, May 2011

The remainder of this paper is structured as follows: in Section 2, we provide an overview of the different types of generalization that have been applied in EBMT. In Section 3, we introduce the two systems which we used for our experiments. In Section 4, we introduce our experimental data set as well as our approach. We then present the results of our experiments and a discussion thereof. In Section 5, we give an overview of the issues which we tackled and offer an outlook on future research questions.

2 Related Work

When compiling generalized templates, there is a risk of replacing too many parts of an SL–TL pair with variables. To avoid this risk of overgeneralization, generalized templates are usually restricted to certain categories of words. Common candidates for generalization are content words, as replacing them with other content words does not affect the grammar of the sentence. Semantic generalization was explored by Kitamura and Matsumoto (1995). Kaji et al. (1992) applied semantic constraints to their approach to syntactic generalization. Pure syntactic generalization was performed by Güvenir and Tunc (1996).

Cicekli and Güvenir (2001) generalized over sequences of words. The underlying assumption is that given two SL–TL sentence pairs, if the two SL sentences have certain word form sequences in common, the corresponding TL sentences are expected to exhibit the same similarities among each other. The similar parts of the SL sentences are then assumed to be translations of the similar parts of the TL sentences, and the same applies for the differing parts.

In the following section we describe the two EBMT systems which we used for our experiments. Both systems started out as purely lexical EBMT systems, i. e., they did not make use of generalized templates. We describe their original approach and explain how they were extended. For the first system, *Marclator*, the extension consists of applying generalization over function words. The second system, *CMU-EBMT* makes use of semantic and, to some extent, syntactic generalization.

Category	Example
determiner	*den*
personal pronoun	*euch*
demonstrative pronoun	*jenem*
possessive pronoun	*seine*
interrogative pronoun	*welch*
indefinite pronoun	*andere*
relative pronoun	*denen*
preposition	*abseits*
coordinative conjunction	*aber*
subordinative conjunction	*falls*
cardinal numeral	*eins*
numeric expression	*neunundneunzig*
auxiliary/modal verb	*darf*
punctuation	*!*

Table 1: German Marker categories and examples

3 Syntactic and Semantic Generalized Templates

3.1 EBMT at DCU: *Marclator*

Marclator was developed at Dublin City University (DCU) and is part of the *MaTrEx* architecture (Stroppa and Way, 2006).[2] The system does not apply the greedy matching strategy typical of many EBMT systems. Instead, it segments both the training and the test data into chunks. Chunking is based on the Marker Hypothesis (Green, 1979). This is a psycholinguistic hypothesis stating that every language has a closed set of elements that are used to mark certain syntactic constructions. The set of elements includes function words and bound morphemes, such as *-ing* as an indicator of English progressive-tense verbs and *-ly* as an indicator of English adverbs.

The *Marclator* chunking module solely considers function words as indicators of chunk boundaries. Each function word (subsequently called *Marker word*) triggers the opening of a new chunk, provided that the preceding chunk contains at least one non-Marker word; e. g., *He was* | ***on** the bus*. For English and German, this leads to left-marking chunks, with the exception that a few German prepositions can also function as postpositions, in which case they are right-marking (e. g., *wegen*: *der Sache **wegen***).

For English, *Marclator* relies on a Marker word list derived from the English monolingual dictionary of the rule-based MT system *Apertium* (Tyers et al., 2010).[3] The German Marker words were ex-

[2] http://www.openmatrex.org/marclator/marclator.html
[3] http://www.apertium.org/

tracted from the *Celex* database.[4] The lists contain a total of 450 Marker words for English and 550 for German. Table 1 lists a sample Marker word for each category. The examples show that entries are included in their inflected forms. Stroppa and Way (2006) found that treating the punctuation marks *! ? , . : ;* as additional Marker elements improved performance in their experiments.

Following the chunking of the training data, *Marclator* performs word and chunk alignment. The system relies on *Giza++* (Och and Ney, 2003) for word alignment. The chunk alignment algorithm is an edit-distance style algorithm in which the distances are replaced by opposite-log conditional probabilities (Tinsley et al., 2008).[5] The recombinator of *Marclator* is a left-to-right monotone recombinator. When translating an input sentence, it first looks for a matching sentence in the example base. If none is found, the sentence is chunked. Each chunk that is not found in the example base is then split into single words. If several TL correspondences for an SL chunk or word are found in the example base, the one with the highest probability is chosen.[6] Thus, for each input sentence, the recombinator outputs a single hypothesis.

A problem inherent in the approach described above is that the chunks of an input sentence often cannot be found in the example base. Since translating a chunk as a whole is likely to yield a better translation than translating it word by word, it is desirable to increase the chunk coverage of a system. Gough and Way (2003) extended the precursor to *Marclator* by including an additional layer of abstraction: they produced generalized chunks from word form chunks by replacing the Marker word at the beginning of a word form chunk with the name of its category, e. g., *of a marathon* → *<PREP> a marathon*. The generalized template extension is not part of the current *Marclator* system. We reimplemented it for our experiments.

3.2 *CMU-EBMT*

The second EBMT system which we used for our experiments is *CMU-EBMT*.[7] The system forms part of *PanLite* (Frederking and Brown, 1996), an MT architecture developed at Carnegie-Mellon University (CMU). It can also be invoked on its own. The system requires a parallel corpus and a bilingual dictionary. Brown (1996) used entries from a commercial bilingual dictionary for his experiments in translation from Spanish to English.

Unlike *Marclator*, *CMU-EBMT* does not require subsentential units to be compiled before the actual translation step. The matching step resembles closely that of a traditional EBMT system: *CMU-EBMT* extracts every substring of the input sentence with a minimum length of two tokens that appears in the SL half of the example base. For each of these fragments, it then identifies the smallest and the largest possible segment in the TL sentence that correspond to it. This is done on the basis of a bilingual dictionary and, optionally, a TL synonym list. Every possible substring of the largest segment that contains at least the minimal segment receives a score. The best alignment is the one with the lowest score. The alignment score is the weighted sum of the values of eight features, which include: the number of SL words with no correspondences in the TL segment, the number of TL words with no correspondences in the SL fragment, the number of SL words with a correspondence in the TL sentence but not in the relevant TL segment, and the difference in length between the SL and the TL segment. Each translation is passed on to the recombination step as long as its score does not exceed five times the length of the SL fragment.

Brown (1999) proposed an extension to *CMU-EBMT* that makes use of semantic and syntactic generalized templates. He referred to the template categories as *equivalence classes*. Examples of semantic and syntactic equivalence classes are given in Table 2. The table shows that class members can in turn contain classes. This is evident from the last line (shown in bold).

The system generalizes both the training and the test set: it recursively replaces words and phrases that are part of an equivalence class with the corresponding class tag. Syntactic classes are applied before semantic classes, and disambiguation numbers are introduced to distinguish between multiple occurrences of the same class tag in a sentence. In the training data, generalization is performed only if a member of a particular equivalence class is found in both the SL and the corresponding TL

[4] `http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC96L14`

[5] Note that both word and chunk alignment involve statistical knowledge.

[6] This is a common procedure for recombinators that do not incorporate a language model.

[7] `http://sourceforge.net/projects/cmu-ebmt/`

Class	Sample member
<religion>	*Christianity – Christentum*
<month>	*December – Dezember*
<fullname-m>	*<firstname-m> <lastname> – <firstname-m> <lastname>*
<fullname-m>	*George Washington – George Washington*
<adj-s>	*affordable – accesible*
<noun-m-p>	*painters – pintores*
<np-m>	*<poss> <noun-m> – <poss> <noun-m>*
<np-f>	*the <noun-f> – la <noun-f>*
<np-f>	***a <color> <noun-f> – une <noun-f> <color>***

Table 2: Semantic and syntactic equivalence classes

sentence. In the case of single-word replacements, the word forms that are replaced are retained as alternatives during the matching process. This does not apply to replacements of more than one word, due to the difference in length to the (single-token) class tags.

In the sentences of the test set, all members of an equivalence class are replaced recursively. The matching process is equivalent to that of the purely lexical *CMU-EBMT* system, with the apparent difference that here, two matching levels – a lexical and a generalized one – exist. Alignment proceeds in the same way as in *CMU-EBMT*. Following this, the rules that were stored during the generalization of the input sentence are applied in reverse so as to transform the generalized TL fragments into word form TL fragments.

4 Experiments and Evaluation

4.1 Our Approach

Our experimental data set consisted of English–German subtitles that were kindly provided to us by a commercial subtitling company. Our corpus contained 1,133,063 subtitles which consisted of on average 8.9 tokens for English and 7.9 for German. For our experiments in translating from English to German, we divided the subtitle data into a training set of 1,130,717 subtitles and a test set and development set of 1173 subtitles each.

Our approach to EBMT consisted of combining the generalized template extensions of the *Marclator* and *CMU-EBMT* systems described in Section 3. This meant building a new system that applies both the DCU and the CMU generalization scheme. Our goal was to see whether our combined system could outperform the two individual systems. For this, we ran an experiment with the combined system as well as one with each individual system. We (re-)implemented the three approaches on top of *Marclator*: we included the word alignment, Marker-based chunking and chunk alignment module of *Marclator*. We also used the *Marclator* recombinator and adjusted it separately for each of the three systems so as to make it capable of dealing with the particular generalization scheme. In summary, we built three systems: *Marclator* with DCU generalized templates (*System 1*), *Marclator* with CMU generalized templates (*System 2*) and *Marclator* with DCU & CMU generalized templates (*System 3*). In what follows, we describe each of these systems along with the baseline systems.

System 1 includes the generalized template extension to *Marclator* that was described in Section 3.1. Recall that *Marclator* is based on Marker words, which are function words. Hence, the extension generalizes over the Marker words at the beginning of Marker-based chunks. We reimplemented it by using the *Marclator* components mentioned above and adding a module that generalizes the aligned SL–TL chunk pairs. We also extended the *Marclator* recombination module: in its original form, the recombination module checks for the presence of matching sentences and word form chunks[8] before reverting to word-by-word translation. We added an additional matching step to follow the chunk matching: in this step, the system replaces the Marker word at the beginning of a chunk by its corresponding Marker tag and searches for the resulting generalized chunk in the example base. Where this attempt fails, the system reverts to word-by-word translation.

The only difference remaining to the approach described in Section 3.1 is that the system of Gough and Way (2004) outputs all possible hypotheses for an input sentence, while the *Marclator* recombinator only outputs the one-best hypothesis. This means that once our system has

[8]We subsequently refer to word form chunks (as opposed to generalized chunks) simply as chunks.

established a generalized chunk match with the SL side of the example base and has extracted the corresponding TL generalized chunk, it has to make a decision as to which Marker word to insert for the Marker tag. For this, it identifies the SL Marker word underlying the SL generalized chunk that was matched. It gathers the word alignment links that contain the SL Marker word and chooses the alignment with the highest frequency, provided that the resulting TL word is also a Marker word.

For example, assume that an SL chunk *i 've finally got* cannot be found in the example base. *System 1* therefore generalizes it to *<PERS_PRON> 've finally got* and extracts the corresponding TL generalized chunk, which is *<PERS_PRON> haben*. The system subsequently searches for a German translation for the SL Marker word *i* (underlying the SL Marker tag *<PERS_PRON>*) in the word alignments. Assuming that it finds *ich*, it produces the TL chunk *ich haben*.[9]

Figure 1, adapted from Armstrong et al. (2006), visualizes the system's training and translation process. The numbers attached to the arrows (I to IV) specify the matching order.

System 2 incorporates the CMU semantic and syntactic equivalence classes described in Section 3.2. Of the 81 classes for the language pair English–German that were provided to us by the developer of the *CMU-EBMT* extension, the majority are semantic classes. The classes contain a total of 5545 replacement rules. Recall that a replacement rule specifies an equivalence class tag and an SL–TL pair whose two halves may be replaced by the tag. Unlike the original implementation (*CMU-EBMT*), *System 2* has *Marclator* at its core, which means that it relies on the word alignment, chunking, chunk alignment and (extended) recombination module of *Marclator*. We implemented an additional module that generalizes Marker-based chunk pairs on the basis of the CMU generalized templates.

System 3 combines Systems *1* and *2*. Accordingly, it generalizes over DCU Marker words as well as CMU semantic and syntactic equivalence classes. Like Systems *1* and *2*, the system has *Marclator* at its core. This makes it possible to directly compare the effectiveness of the generalization schemes. However, the DCU and the CMU generalization schemes are not mutually exclusive. There are a number of overlaps, i. e., the CMU classes contain 50 words that are also Marker words for English (e. g., *after*, *and*, *before*), and 19 for German (e. g., *aber*, *allen*, *er*). We prompted the system to generalize over the Marker words first, thereby giving preference to the DCU scheme in case of overlaps.

[9]Note that this translation is deficient. We discuss the problems inherent in the approach of *System 1* in Section 4.3.

System	BLEU	NIST	METEOR
1	0.1274	**4.3948**	**0.4052**
2	0.1269	4.3815	0.4047
3	**0.1277**	4.3937	0.4051
Marclator	0.0995	4.2411	0.3990
OpenMaTrEx	0.2763	5.7880	0.4914
Moses	0.2709	5.7472	0.4854

Table 3: Evaluation scores

Baselines: We established three baselines: *Marclator*, *OpenMaTrEx* (Dandapat et al., 2010) and *Moses* (Koehn et al., 2007). The *Marclator* baseline was the purely lexical system described in Section 3.1. For the *Moses* baseline, we used the default system included in *OpenMaTrEx*. The system uses a 5-gram language model and modified Kneser-Ney smoothing. Training is performed according to the default options and thus includes tuning via MERT (Och, 2003). In addition, a lexicalized reordering model is learnt. The *OpenMaTrEx* baseline system makes use of EBMT chunk pairs from *Marclator* and SMT phrase pairs from *Moses*. We used the default configuration, which includes a 5-gram language model with modified Kneser-Ney smoothing and tuning via MERT. We included the optional binary feature that records whether a phrase pair is an EBMT chunk pair or not. To train the language models for *Moses* and *OpenMaTrEx*, we used the TL side of the training data.

4.2 Results of the MT Systems

Table 3 shows the results of our experiments. The best of our systems (Systems *1* to *3*) with regard to each of the three evaluation metrics (BLEU, NIST and METEOR) is shown in bold. The table shows that there was no agreement among all three metrics as to which system performed best: *System 3* performed best according to BLEU, while *System 1* performed best according to NIST and METEOR. The three systems outperformed the lexical baseline system *Marclator* according to all three

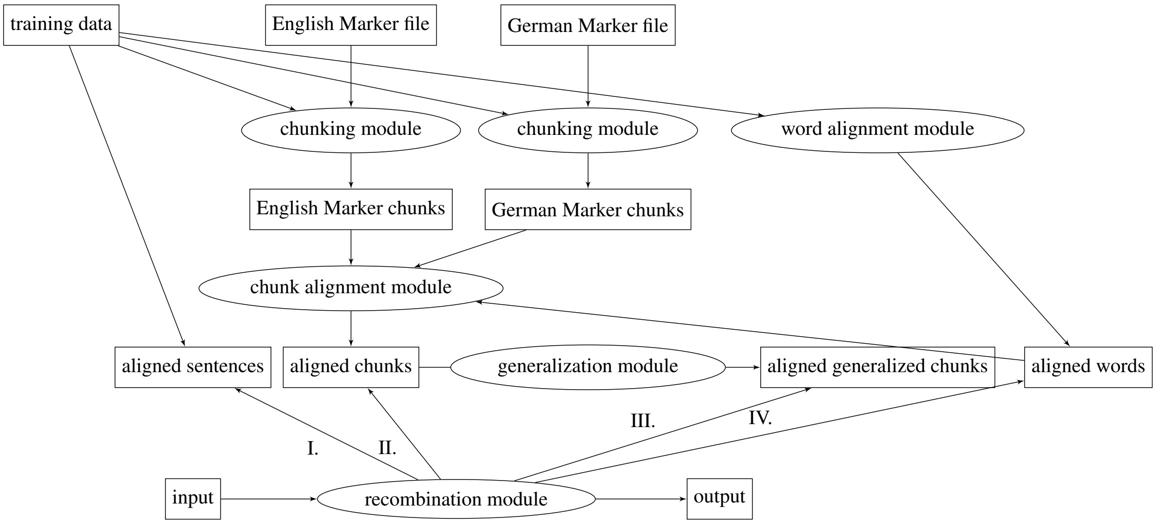

Figure 1: *System 1*: training and translation process

metrics.[10] We measured statistical significance by bootstrap resampling (Koehn, 2004) on BLEU.[11] The improvement of *System 3* over *System 2* is statistically significant, while the improvement of *System 3* over *System 1* is not. The improvements of Systems *1*, *2* and *3* over the baseline *Marclator* system are all significant, as are the improvements of the baseline *OpenMaTrEx* and *Moses* system over Systems *1* to *3*.

4.3 Chunk Coverage and Chunk-Internal Boundary Friction

The evaluation results in Table 3 show that our generalized EBMT systems achieved higher scores than the lexical EBMT system *Marclator*. This observation supports earlier findings according to which EBMT systems benefit from generalized templates. We believe that it is reinforced by the performance results of *System 1* and *System 2*: *System 1* performed better than *System 2* according to all three evaluation metrics. We investigated the generalized chunk coverage of the two systems, i. e., the number of successful generalized chunk matches with respect to the total number of attempts made at matching a generalized chunk.[12] The coverage was 8.26 % for *System 1*. For *System 2*, it was 2.14 %, which is very low. We conclude from this that the higher generalized chunk coverage of *System 1* was the reason why this system performed better than *System 2*.

Table 3 also shows that combining *System 1* and *System 2* into *System 3* did not yield a clear improvement over the individual performances of these two systems. We think that this is due to minor differences in the way in which chunks are generalized in our systems as well as to overlaps in the generalization schemes: recall that the two schemes have certain class members in common. The results might also indicate that *System 3* overgeneralized.[13] However, we think that this explanation is not valid in our case: we demonstrated that the CMU generalization scheme led to a low generalized chunk coverage in *System 2*. Hence, it also did not contribute many generalized chunks to the translation process of *System 3*.

We believe that the low generalized chunk coverage of *System 2* demonstrates the problem inherent in the use of semantic word classes, which form the majority of the CMU equivalence classes. The classes are very specific; many of them (e. g., *city, company, country*) have proper name members. On average, each class contains 69 members. To improve the generalized chunk coverage, this number would have to be increased.

When investigating the output of *System 1*, we observed one major source of errors, which we call *chunk-internal boundary friction*. Boundary friction is normally caused by the juxtaposition of two separate translation units that do not agree in gram-

[10]In experiments carried out on half of the data, *System 1* performed best according to BLEU, while *System 2* performed best according to NIST and METEOR. Systems *1* to *3* also performed better than the *Marclator* baseline.

[11]An approximate randomization with 500 shuffles was performed for the significance tests. All further statements about significance refer to a significance level of α=5 % and to BLEU.

[12]Recall that a generalized chunk match is attempted after every unsuccessful word form chunk match.

[13]We mentioned the risk of overgeneralization in our introduction.

matical case. With the introduction of Marker-based templates, it can also take place *within* a single chunk, i. e., when a Marker word is inserted that does not accommodate the grammatical properties of the rest of the chunk. In the case of English–German translation, inserting TL Marker words context-insensitively (as is done in *System 1*) is error-prone: due to the morphological richness of German, an English Marker word can correspond to multiple word forms of the same lemma on the German side. For example, the English Marker word *are* can be translated into the German Marker words *bist*, *sind* and *seid*. Example 2 shows an English input sentence and the corresponding German output sentence, translated by *System 1*. The section where chunk-internal boundary friction occurred is shown in bold.

(2) *are you sure that superman was hypnotized last night ? –* ***sind du*** *sicher dass das superman war hypnotisiert gestern nacht ?*

To translate the English sentence in (2) into German, our system made use of a German generalized chunk *<AUX> du sicher*. It then instantiated *<AUX>* with *sind*. *sind* is a German verb in the first or third person plural, while *du* is a second-person singular pronoun and the subject of the sentence. In German, the subject and the verb of a sentence have to agree in person and number. Therefore, the combination of *du* and *sind* is grammatically incorrect.

Table 3 also shows that our EBMT systems performed much worse than the baseline systems *Moses* and *OpenMaTrEx*. We think that the performance gap is largely due to the recombination module of *Marclator*: the recombinator is monotone in nature and outputs only the one-best hypothesis. No language model is applied. Both *OpenMaTrEx* and *Moses* apply a language model for hypothesis recombination. We believe that it is essential for an EBMT system to make use of a language model to reward output sentences that are more fluent with respect to the TL.

In summary, our combined system did not perform significantly better than the two individual systems according to all three evaluation metrics. However, all three generalized EBMT systems outperformed the purely lexical *Marclator* baseline.

5 Conclusion

In this paper, we reported the results of experiments in combining two existing EBMT systems that rely on generalized templates. The combined system did not yield a significant improvement in translation quality compared to the individual performances of the two systems; it still exhibited a low generalized chunk coverage. However, our generalized EBMT systems consistently outperformed the lexical EBMT baseline. This shows that generalized templates are advantageous to an EBMT system's performance.

We demonstrated that it is more difficult to achieve a high generalized chunk coverage with semantic generalized templates than with generalized templates based on function words. Semantic generalized templates have the advantage that they do not interfere with the grammar of a sentence. In contrast, generalized templates based on function words are relatively easy to compile. However, we showed that a system which relies on such templates can suffer from chunk-internal boundary friction. In our English–German experiments, it occurred when German preposition or auxiliary verb Marker tags were instantiated. To reduce the chunk-internal boundary friction problem, we are currently developing an algorithm that context-sensitively instantiates TL Marker tags by using a language model. We plan to incorporate it into our generalized template extension of *Marclator*.

Acknowledgements

We thank Ralf Brown for making available his English–German equivalence classes from the *CMU-EBMT* system.

References

Armstrong, Stephen, Declan Groves, Marian Flanagan, Yvette Graham, Bart Mellebeek, Sara Morrissey, Nicolas Stroppa, and Andy Way. 2006. The MaTreX System: Machine Translation Using Examples. In *TC-STAR OpenLab Workshop on Speech Translation*, pages not numbered, Trento, Italy.

Brown, Ralf D. 1996. Example-Based Machine Translation in the Pangloss System. In *COLING-96: The 16th International Conference on Computational Linguistics, Proceedings*, pages 169–174, Copenhagen, Denmark.

Brown, Ralf D. 1999. Adding Linguistic Knowledge to a Lexical Example-based Translation System. In *8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 99)*, pages 22–32, Chester, England.

Cicekli, Ilyas and Halil Altay Güvenir. 2001. Learning Translation Templates from Bilingual Translation Examples. *Applied Intelligence*, 15(1):57–76.

Dandapat, Sandipan, Mikel L. Forcada, Declan Groves, Sergio Penkale, John Tinsley, and Andy Way. 2010. OpenMaTrEx: A Free/Open-Source Marker-Driven Example-Based Machine Translation System. In *Proceedings of IceTAL*, pages 121–126, Reykjavík, Iceland.

Frederking, R.E. and R.D. Brown. 1996. The Pangloss-Lite Machine Translation System. In *Expanding MT Horizons, Proceedings of the Second Conference of the Association for Machine Translation in the Americas*, pages 268–272, Montreal, Quebec, Canada.

Gough, Nano and Andy Way. 2003. Controlled Generation in Example-Based Machine Translation. In *Proceedings of MT SUMMIT IX*, pages 133–140, New Orleans, USA.

Gough, N. and A. Way. 2004. Robust Large-Scale EBMT with Marker-Based Segmentation. In *TMI-2004 Conference: Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 95–104, Baltimore, Maryland, USA.

Green, T. 1979. The Necessity of Syntax Markers. Two Experiments with Artificial Languages. *Journal of Verbal Learning and Behavior*, 18:481–496.

Groves, Declan and Andy Way. 2005. Hybrid Example-Based SMT: the Best of Both Worlds? In *ACL-05: Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond, Proceedings of the Workshop*, pages 183–190, University of Michigan, Ann Arbor, Michigan, USA.

Güvenir, H. A. and A. Tunc. 1996. Corpus-Based Learning of Generalized Parse Tree Rules for Translation. In *Proceedings of the 11th Biennial Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence (AI'96)*, pages 121–132, London, UK.

Kaji, Hiroyuki, Yuuko Kida, and Yasutsugu Morimoto. 1992. Learning Translation Templates from Bilingual Text. In *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, COLING-92*, pages 672–678, Nantes.

Kitamura, Mihoko and Yuji Matsumoto. 1995. A Machine Translation System Based on Translation Rules Acquired from Parallel Corpora. In *International Conference: Recent Advances in Natural Language Processing, Proceedings*, pages 27–36, Tzigov Chark, Bulgaria.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-based Translation. In *HLT-NAACL 2003*, pages 48–54, Edmonton, Canada.

Koehn, P., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic.

Koehn, Philipp. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain.

Nagao, Makoto. 1984. A Framework of a Mechanical Translation between Japanese and English by Analogy Principle. In *Proceedings of the International NATO Symposium on Artificial and Human Intelligence*, pages 173–180, New York, USA.

Och, Franz Josef and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Och, Franz Josef. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *41st Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 160–167, Sapporo Convention Center, Japan.

Stroppa, Nicolas and Andy Way. 2006. MaTrEx: DCU Machine Translation System for IWSLT 2006. In *Proceedings of IWSLT 2006*, pages 31–36, Kyoto, Japan.

Tinsley, John, Yanjun Ma, Sylwia Ozdowska, and Andy Way. 2008. MATREX: the DCU MT System for WMT 2008. In *ACL-08: HLT: Third Workshop on Statistical Machine Translation, Proceedings of the Workshop*, pages 171–174, Columbus, Ohio, USA.

Tyers, F.M., F. Sánchez-Martínez, S. Ortiz-Rojas, and M.L. Forcada. 2010. Free/open-source resources in the Apertium platform for machine translation research and development. *The Prague Bulletin of Mathematical Linguistics*, 93:67–76.

Way, Andy. 2001. Translating with Examples. In *MT Summit VII: Workshop on Example-Based Machine Translation, Proceedings of the Workshop*, pages 66–80, Santiago de Compostela, Spain.