Phrase-based Machine Translation in a Computer-assisted Translation Environment

Michel Simard **Pierre Isabelle**
National Research Council Canada
283 Taché Blvd., Gatineau (Québec), Canada J8X 3X7
FirstName.LastName@nrc.ca

Abstract

We explore the problem of integrating a phrase-based MT system within a computer-assisted translation (CAT) environment. We argue that one way of achieving successful integration is to design an MT system that behaves more like the translation memory (TM) component of CAT systems. This implies producing MT output that is consistent with that of a TM when high-similarity material exists in the training data; it also implies providing the MT system with a component that is capable of filtering out machine translations that are less likely to be useful. We propose solutions to both problems, and evaluate their impact on three different data sets. Our results indicate that the proposed approach leads to systems that produce better output than a TM, for a larger portion of the source text.

1 Introduction

While much research effort has been devoted to finding ways of combining the strengths of human and machine translation (see Foster (2002) for an overview), in the end, many human translators still find MT technology to be unhelpful when it comes to producing high-quality translations. As a result, MT is not yet well-established within computer-assisted translation (CAT) environments, at least when compared to the much simpler technology of translation memory (TM).

While it is tempting to view TM as a simplistic form of example-based MT, or a variation on phrase-based MT (or, conversely, to see phrase-based MT as a natural evolution of TM), TM has some notable advantages over most data-driven MT systems. The most obvious is its ability to translate predictably and (near-) perfectly any input that it has seen previously. Another quality of TM is its ability to find approximate matches and to let the user adapt system behavior to his/her own tolerance to errors by fixing the similarity threshold on such matches; in other words, TM's benefit from a highly effective confidence estimation mechanism.

If machine translation is to succesfully integrate the CAT environment, it should begin by catching up with TM on these aspects. We argue that this requires two things: (1) the MT system should behave more like a TM in the presence of high-similarity matches. In practice, this can be achieved by combining the two technologies, i.e. by building a combination MT system that incorporates a TM component. And (2) just like existing TM systems, the combined MT system should provide the user with means to filter out translations that are less likely to be useful.

It has sometimes been proposed (see e.g. Heyn (1996)) that MT should be used within a CAT environment only when the TM fails to retrieve something useful. Unfortunately, this has the effect of relegating the MT system to the task of translating only the sentences that are most unlike previously seen ones. For data-driven systems, this turns out to mean translating only the "harder" sentences and missing the chance to do a better job than the TM. The reason why MT is often treated as a last resort lies in the fact that translators tend to see its performance as unpredictable and, as a result, overly likely to waste their time. In other words, to be accepted

as a useful tool by human translators, an MT system needs its own way of determining whether its own output is likely to be useful to the translator.

Because MT systems of the statistical phrase-based variety have a lot in common with TM systems, we take them as a starting point. In what follows, we propose relatively simple ways of attaining the two above goals: Section 2 deals with combining TM with phrase-based MT, and Section 3 discusses estimating translation usefulness. Our experiments and results are presented in Section 4.

2 Combining Machine Translation and Translation Memory

In this section, we examine the question of how to combine a TM with a phrase-based MT system. Our goal is to obtain an MT system that is capable of taking advantage of exact or close matches in the TM.

The methods described here assume a standard phrase-based SMT system (Koehn et al., 2003) employing a log-linear combination of feature functions. Unfortunately, there are no such "standard" TM systems; therefore we had to construct our own, which we now describe before discussing combination strategies.

2.1 Translation Memory

At the core of a TM is a database of existing translations: pairs of source-language and target-language segments of text which are mutual translations. Typically, these text segments are complete sentences. Given a new segment of source-language text to be translated, the system searches its database for an exact or approximate ("fuzzy") match. If such a match is found, its target-language version is proposed to the user, who is then free to reuse it, modify it or discard it. Optionally, the resulting (human-)translation is fed back to the system and stored into the database.

We simulate the translation memory functionality of CAT environments with a collection of programs we call TMem. Given a corpus of existing translations, in the form of source-target pairs of sentences, and a new source-language sentence to be translated, which we call the *query*, TMem computes the word-based Levenshtein distance between the query and each source-language sentence in the corpus, and retains the source-target pair from the corpus with the smallest distance. This is admittedly very inefficient (although manageable if one resorts to simple optimizations and some parallelization), and while Levenshtein distance may not be the *nec plus ultra* in TM technology, this is possibly compensated by the fact that TMem performs an exhaustive search. Additionally, when there are ties in the translation memory, with alternative translations, we use an IBM Model 2 component to pick the translation that is most likely given the source.

In practice, CAT systems allow the user to set a threshold on source similarity, which prohibits the TM component from outputing irrelevant translations when the corresponding source segment is too different from the query. The higher the threshold is set, the better the quality of the proposed translations. Of course, there is no magic: when raising this threshold, users are simply trading recall for precision: the increased quality comes at the expense of decreasing the system's coverage, i.e. the proportion of queries for which the system does propose translations.

To perform this sort of filtering, TMem relies on a length-normalized variant of Levenshtein distance:

$$sim(q,s) = \max(0, 1 - \frac{levenshtein(q,s)}{length(q)})$$

where q is the query and s is the best-matching source-language segment in the corpus. We refer to TMem's user-set threshold on the value of sim as α (alpha): if $sim(q,s) \geq \alpha$, then TMem outputs the translation of s, otherwise it outputs nothing.

2.2 Related Work

There is a rapidly growing body of work on MT system combination (see e.g. Callison-Burch et al. (2009)), and many of the methods proposed in the literature could be applied to the specific task of combining a TM with a phrase-based MT system.

Somewhat parallel to this, a number of authors have examined specifically the MT-TM tandem. Much work in this line actually aims at producing better MT systems, as opposed to integrating MT into a CAT environment. For instance, Vogel et al. (2004) present a SMT system which incorporates a translation memory component. The system outputs

exact matches from the TM without further processing. Automatic "repairs" are performed for matches that display a single "error" (insertion, deletion or substitution): this operates essentially like a one-step greedy modification on the TM target.

Leplus et al. (2004) show how a translation memory equipped with a minimal hand-built alteration mechanism for numbers, etc., can be quite successfully used as a MT system for repetitive texts such as weather reports.

The *Dynamic Translation Memory* (DTM) method (Biçici and Dymetman, 2008) also aims at improving the output of a phrase-based SMT system using a TM: Given a new sentence q to be translated, they:

1. Find the best matching pair for q in the TM: $\langle s, t \rangle$;
2. Identify the longest common subsequence between q and s: P_s;
3. Using word-alignment, identify the corresponding subsequence in t: P_t;
4. Dynamically add the phrase pair $\langle P_s, P_t \rangle$ to the translation system's phrasetable

They then translate as usual. This strategy crucially depends on an essential characteristic of the MT system in which it is implemented: the MATRAX phrase-based SMT system (Simard et al., 2005) can handle non-contiguous phrase pairs, i.e. phrases with "gaps" both in the source and the target. This makes it possible in step 2 above to build a phrase pair from *subsequences* (not substrings) which covers as much commonalities as possible between q and s.

Then there is also some work that aims at improving TM systems, using MT technology.

Schäler (2001) proposes the idea of a (syntactic) phrase-based translation memory that would be able to mix-and-match phrases from different TM matches, in order to piece together a proposal for a previously unseen sentence, EBMT fashion. But in practice, this is more a step for TM in the direction of MT.

Simard and Langlais (2001) evaluate the potential of complementing a translation memory with a phrasetable. Phrases are proposed to the user based on a maximum cover of the input, taking into account a rudimentary translation model. A step forward in the same direction is proposed by Gotti et al. (2005), who suggest restricting phrases to syntactic chunks or treelets.

2.3 Combination Strategies

In what follows, we propose two different strategies which we believe are better suited to the particular characteristics of TM and phrase-based MT.

2.3.1 System selection: β-combination

The simplest form of MT-TM combination is probably one in which either the MT or the TM output is produced, depending on the context. This strategy has been proposed for combining multiple MT systems, for example in Nomoto (2004). Many factors can be taken into account when deciding which system is best for a given input; this decision process can be viewed as a standard classification task, and many standard machine learning methods can be applied.

An extremely simple, yet effective approach is to base the decision solely on the similarity between the query and the closest match from the translation memory, as proposed by Vogel et al. (2004): above a given similarity threshold β, the combined system outputs the translation from the TM, otherwise it produces the MT output. We call the systems based on this combination strategy β-combinations. The value of β can be optimized to maximize some measure of translation quality (e.g. BLEU or WER) on a held-out sample.

2.3.2 TM-based Translation Feature Functions

As mentioned earlier, the method of Biçici and Dymetman (2008) is not directly applicable to standard phrase-based systems, because it relies crucially on discontiguous phrases. However, we can propose a very similar approach, in which multiple phrases are extracted from the best TM match and fed to the MT system. More precisely, for each input sentence q, we find the single best matching pair $p = \langle s, t \rangle$ from the TM, using the TMem program. We then compute the set T_p of all admissible phrase pairs from p as in Och & Ney (2004), but without limiting phrase size. In particular, this means that if we find an exact matching sentence in the TM, then

this is presented as a "pretranslated phrase" to the MT system, regardless of its size.

The content of this *TM-based phrasetable* is used at decoding time as an additional source of phrases, and its weight in the loglinear model is determined by MERT on a heldout set, as usual. (Chen et al. (2009) propose a similar method for combining the output of multiple MT systems.)

We can use a similar trick to provide the MT system with a *TM-based language model*: use the target-language portion of the single best match from the TM to train a sentence-specific language model, which we use as an additional feature function in the model.

Finally, while the MT system may have all the right phrases to put together a perfect match from the TM, it may opt for alternate phrase translations, or simply order the phrases differently. One way to coerce the system into generating something that resembles a TM match as closely as possible is to introduce a *TM-match similarity* feature function, which gives better scores to translations whose word-order is more similar to a given target-language sentence. This can be achieved by means of distance measures such as Levenshtein distance (as in word-error rate) or n-gram precision (as in the BLEU metric).

In practice, such similarity feature functions are extremely costly to use at decoding time: in our experience, in spite of numerous optimizations, using either one of these will typically multiply decoding times by a factor of 10. Fortunately, we have found that it is possible to achieve almost similar results by applying them only at a rescoring stage, where application is restricted to a list of n-best translations provided by the decoder, and therefore much less costly. In our experiments, we actually use three TM-based similarity features concurrently: Levenshtein distance, 1-gram precision and bigram precision.

3 Target Usefulness Estimation

As mentioned earlier, CAT software usually allows the user to set a threshold on similarity between the query and TM matches. The implicit assumption is that source-language similarity is a reliable predictor of the usefulness of the target-language segment for the user.

We want to provide an MT system with a similar mechanism. We propose to approximate usefulness by the similarity of the target translation with a reference translation. While there are different ways to measure this, the Levenshtein-based sim function of Section 2.1 is appealing because it is simple to compute and has an intuitive interpretation. Furthermore, when applied to target-language translations, it is related in obvious ways to the well-known *word-error rate* (WER) metric: $sim(t, r) = 1 - WER(t, r)$, where r is a reference (correct) translation and t is the translation whose usefulness we wish to estimate.

Estimating target similarity is quite obviously related to work on confidence estimation for machine translation. As proposed for the latter task (Quirk, 2004; Gandrabur et al., 2006), we take a supervised machine learning approach to the problem: the idea is to base a target quality estimation (QE) function on a variety of features of the input and output texts, and learn output values from training data annotated with target similarity values.

We consider the following input features:

- length of the source input q, best-matching TM source s, corresponding target t and MT output p;
- probabilities $P_{SLM}(q)$, $P_{SLM}(s)$, $P_{TLM}(t)$ and $P_{TLM}(p)$, according to N-gram language models, trained on the source ("SLM") or target ("TLM") portions of the translation memory;
- language model probability ratios $P_{SLM}(q)/P_{TLM}(t)$ and $P_{SLM}(q)/P_{TLM}(p)$;
- various measures of similarity between q and s, and between t and p: Levenshtein distance, longest common substring, n-gram precision (for n =1 to 4)
- IBM Model 2 estimates of $P(q|t)$, $P(t|q)$, $P(s|t)$, $P(t|s)$, $P(q|p)$ and $P(p|q)$. Here again, IBM models were trained on the source and target portions of the translation memory.

Different types of learners can be applied to this task, the most obvious being least-squares linear

regression. However, in our experience, the most stable results were obtained with regression support vector machines. We used the **libsvm** implementation of ϵ-regression SVM's, with all the default settings provided through the **e1071** R interface (Meyer, 2001).

4 Experiments

4.1 Corpora

Our experimental data consists in French-English bilingual corpora. In all our experiments, we assumed English to be the source language and French to be the target. All experiments were performed on three distinct data sets, namely the *Hansard*, *Acquis*, and *Europarl* corpora.

Acquis The *Acquis* corpus is the European Community's "Acquis communautaire", prepared and distributed by the Community's *Joint Research Centre* in Ispra, Italy (Steinberger et al., 2006), version 2.2. We chose this corpus because it is rather technical, with lots of internal repetition. This makes it a good candidate for TM. While the corpus is available in over 20 languages, only the English and French versions were used here.

The data had to be: converted from XML to plain line-for-line sentence alignment format; tokenized and re-segmented into sentences; re-aligned at the sentence level, using a variant of the Gale & Church method (the provided alignment was at the paragraph level); and lowercased.

Even after re-alignment, some lines were excessively long and caused problems in the MT system, so we kept only those pairs of sentences shorter than 800 characters (that's still a healthy 150-200 words...); in the process, we also removed pairs in which one sentence was empty.

Finally, the data was split into three parts: *train*, *dev* and *test*. The last two contain 3000 pairs of sentences each, which leaves close to 330k sentence pairs in *train*.

Europarl The *Europarl* corpus is a collection of text from the proceedings of the European Parliament. We used the French and English versions of this corpus as prepared by Philipp Koehn (2005) for training SMT systems. This is a well-known dataset that is commonly used in MT experiments. The Europarl corpus contains close to 1.3 million pairs of sentences; again, we split this data into distinct sets *train*, *dev* and *test*. The two latter each contain 2000 sentence pairs, while *train* contains the remaining sentence pairs (approx. 36 million words of English).

Hansard This is also a well-known dataset in MT. In this case, however, the data was collected independantly, and contains not only Canadian parliamentary debates, but also proceedings of the Canadian Senate and of various parliamentary committees, spanning a 12-year period. Like the Acquis corpus, this corpus went through clean-up, tokenization, segmentation into sentences, sentence-level alignment and lowercasing. Here the *dev* and *test* sets each contain approximately 1500 sentences, while the *train* set contains approximately 5.2 million sentence pairs (just over 100 million English words).

4.2 System Configuration

All instances of the MT system discussed here were essentially trained in the same way. Distinct systems were trained on each of the three corpora. Phrase tables and language models were extracted from each corpus's *train* sets. HMM translation models were used to perform word alignments on the training corpus, which were symmetrized by the usual "diag-and" algorithm prior to phrase extraction. Phrases were limited to a maximum of 8 words. In addition to raw joint translation probabilities, various smoothed conditional probabilities were used as distinct phrase feature functions (Foster et al., 2006). The target-language models were 4-gram models with Kneyser-Ney smoothing.

The *dev* sets were then used to optimize the model's decoding and rescoring parameters. In addition to the decoding feature functions, rescoring also relied on IBM-Model based features, plus some nbest-post features, some features that check for mismatched parentheses, quotes etc. Parameter optimization was performed using minimum error-rate training (MERT) on BLEU scores (Och, 2003).

4.3 Translation Results

Table 1 presents the performance of the main systems we tested on each corpus' *test* set. Results are

presented both in terms of BLEU scores and word error-rate (WER).

The first line shows the results of using only a TM, namely our TMem program. These results reveal some fundamental differences between our different corpora with regard to internal repetition. Europarl is a typical example of a corpus for which TM's are mostly useless; as it contains very little sentence repetition, TMem's performance is extremely low. In comparison, TMem performance on the Hansard corpus is surprisingly high, revealing an unexpected amount of repetition in that corpus. Upon analysis, this is explained by the presence of much procedural or formulaic material (session opening, closure, etc.) and also by the volume of material (over 100M words). Finally, and as pointed out earlier, the Acquis corpus is inherently very repetitive, and TMem displays its strongest performance there.

Internal repetition also seems to affect MT results positively, as can be seen on line 2 of Table 1: the performance of our baseline MT system is much better on the Hansard than on Europarl, and reaches an unusually high 56.8 BLEU on the Acquis.

The following lines display the results of combining our MT system with TMem using a β-combination (line 3) and TM-based feature functions as discussed in section 2.3.2 (line 4). On Europarl data, neither approach yields any visible gains. This confirms our intuition that a combination with a TM can only be productive if the TM itself can extract useful material. On the Hansard and Acquis, however, both strategies brings significant performance improvements.

As line 5 indicates, the overall best performance is obtained when the β-combination and TM-based feature functions are used together. On the Hansard, we then gain 1.8 BLEU (1.9 WER) over the baseline while on the Acquis, the gain reaches 2.6 BLEU (2.9 WER).

In these experiments, parameter β was set to 1; in other words, the TM output is only used as such in the case of a 100% match. Early experiments revealed that automatically optimized values were close to $\beta = 1$ and did not lead to significantly better translations. However, the decision was actually based on a more rhetorical argument: when source-similarity is below 1, the target-language material coming from the TM cannot generally be expected to be a proper translation of the query, since it is known to be the translation of something else. In an MT perspective, producing output which is *known* to be wrong is somewhat disturbing. Therefore, we chose to restrict the use of the TM to exact matches only.

4.4 Quality Estimation Results

The effect of our QE component (Section 3) can be seen in Figure 1, which shows the tradeoff between translation quality and source text coverage as we modify the α threshold on system output. Text coverage was measured as the proportion of source-language words for which the system did propose a translation. Output quality was measured only on proposed translations (i.e. we ignored sentences for which the system did not propose a translation). For the TMem system, thresholding was done on the sim function (Section 2.1); for MT systems (baseline and combined), it was performed on estimated target quality. The combined MT system (labeled MT+TMem) uses all TM-based feature functions proposed in Section 2.3.2 and a β-combination with TMem output, with $\beta = 1$.

On such graphs, the upper right corner represents an "ideal system", i.e. one that would produce "perfect" translations (BLEU=1) on all of the system's input (coverage=100%). The right end of each curve (coverage=1) depicts the global quality of the corresponding system's output, as reported in Table 1. The shape of each curve is solely determined by the output filtering mechanism, within the limits of the system's performance: the more accurately it predicts output quality, the closer the curve should be from the "ideal" upper-right corner.

For Europarl data, there is again no visible difference between the baseline and combined MT systems, as can be seen in the top graph in Figure 1. As expected, MT systems perform noticeably better than TMem in this context. On Hansard and Acquis data, however (middle and bottom graphs), we see that the gains over the baseline systems reported in Table 1 can be observed at almost all levels of input coverage. The apparent instability of all system's performance in the low-coverage area (left end of the curves) can be explained in part by the fact that the samples from which we estimate BLEU scores

System	Europarl BLEU	Europarl WER	Hansard BLEU	Hansard WER	Acquis BLEU	Acquis WER
TMem	5.9	0.840	21.0	0.702	36.3	0.556
MT baseline	31.5	0.592	42.0	0.497	56.8	0.364
MT + TMem($\beta = 1$)	31.5	0.592	43.7	0.482	57.6	0.358
MT + TM features	31.2	0.599	43.4	0.482	59.0	0.348
MT + TM features + TMem	31.2	0.600	43.8	0.478	59.4	0.335

Table 1: Overall System Performance

are inherently small in that area.

The most striking feature of these curves is that for all three datasets, the combined MT systems always perform better than the translation memory: at equal coverage, they provide better translations (as measured with the BLEU metric, at least); and at equal quality, they propose translations for a larger portion of the input.

5 Conclusions

The work presented in the previous pages focused on the problem of integrating a phrase-based MT system within a CAT environment. We have proposed an approach in which the MT system emulates the behavior of a TM. Along this line, we have examined two complementary ways of combining phrase-based MT and TM technology: one that simply selects the most appropriate component (TM or MT) given the context, and one that allows the phrase-based system to actively exploit the most similar material identified by the TM, via TM-based feature functions. In our experiments, both of these approaches lead to significant gains in MT quality when close or exact matches for the input segment were present in the training material; optimal results were obtained when both were deployed.

We have also shown how a fairly simple, machine-learned "quality estimation" layer can be used to filter out machine translations that are the less likely to be useful to the translator. Because this component is designed to produce a real-valued quality estimate, the user can set the threshold on output filtering to fit his/her own tolerance to machine errors. The resulting device achieves the purpose of singling out the best translations, to a degree where the MT system always produces better translations than a TM, whatever the degree of coverage sought.

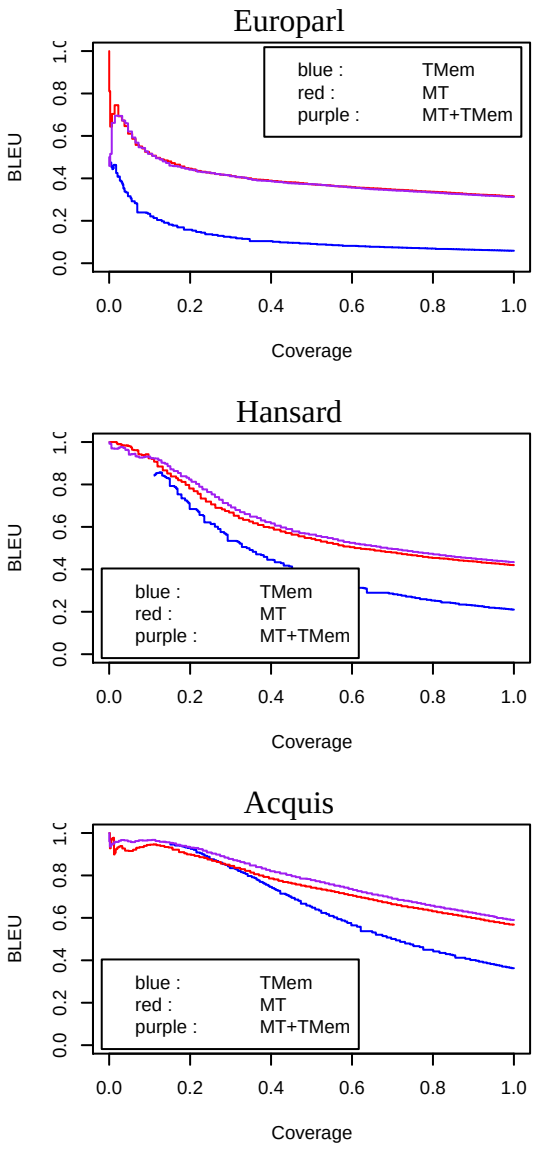

Figure 1: Tradeoff between input text coverage and translation quality.

The quality estimation component could obviously be improved in different ways: many obvious additional features could be exploited, most notably system internal features such as the number and size of phrases used in the translation, presence of out-of-vocabulary words in the input, etc. In our experiments, this component was also typically trained on very little data, and presumably better predictions could be expected with more substantial training sets. Yet, it seems inappropriate to invest massively in the current direction, because in a real-life setting, the QE component would possibly benefit from something which we did not have easy access to in the course of this study: real user feedback. Quirk (2004) points out that just a small amount of coarse user judgements leads to much more reliable confidence estimations than much larger quantities of automatically annotated data.

References

E. Biçici and M. Dymetman. 2008. Dynamic Translation Memory: Using Statistical Machine Translation to Improve Translation Memory Fuzzy Matches. *Lecture Notes In Computer Science*, 4919:454–465.

C. Callison-Burch, P. Koehn, C. Monz, and J. Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece, March. Association for Computational Linguistics.

Y. Chen, M. Jellinghaus, A. Eisele, Y. Zhang, S. Hunsicker, S. Theison, C. Federmann, and H. Uszkoreit. 2009. Combining multi-engine translations with Moses. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 42–46, Athens, Greece, March. Association for Computational Linguistics.

G. Foster, R. Kuhn, and H. Johnson. 2006. Phrasetable smoothing for statistical machine translation. In *Proceedings of EMNLP 2006*, pages 53–61.

G. Foster. 2002. *Text Prediction for Translators*. Ph.D. thesis, Université de Montréal, may.

S. Gandrabur, G. Foster, and G. Lapalme. 2006. Confidence estimation for NLP applications. *ACM Transactions on Speech and Language Processing*, 3(3):1–29.

F. Gotti, P. Langlais, E. Macklovitch, D. Bourigault, B. Robichaud, and C. Coulombe. 2005. 3GTM: A Third-Generation Translation Memory. In *3rd Computational Linguistics in the North-East (CLiNE) Workshop*, Gatineau, Québec, aug.

M. Heyn. 1996. Integrating machine translation into translation memory systems. In *EAMT Machine Translation Workshop*, page 113, Vienna, Austria.

P. Koehn, F.J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics Morristown, NJ, USA.

P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit*, volume 5.

T. Leplus, P. Langlais, and G. Lapalme. 2004. Weather Report Translation using a Translation Memory. In *Machine Translation: from Real Users to Research: 6th Conference of AMTA*, Lecture Notes in AI #3265, pages 154–163, Washington, sep. Springer.

D. Meyer. 2001. Support vector machines. *R News*, 1(3):23–26, September.

T. Nomoto. 2004. Multi-engine machine translation with voted language model. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics Morristown, NJ, USA.

F.J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

F.J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics Morristown, NJ, USA.

C. Quirk. 2004. Training a sentence-level machine translation confidence measure. In *Proceedings of LREC 2004*, pages 825–828.

R. Schäler. 2001. Beyond Translation Memories. In *Proceedings of MT Summit VIII*, pages 49–55, Santiago de Compostela, Spain, September.

M. Simard and P. Langlais. 2001. Sub-sentential Exploitation of Translation Memories. In *Proceedings of MT Summit VIII*, pages 385–390, Santiago de Compostela, Spain, September.

M. Simard, N. Cancedda, B. Cavestro, M. Dymetman, E. Gaussier, C. Goutte, K. Yamada, P. Langlais, and A. Mauser. 2005. Translating with non-contiguous phrases. In *Proc. of HLT-EMNLP*, pages 755–762.

R. Steinberger, B. Pouliquen, A. Widiger, C. Ignat, T. Erjavec, D. Tufis, and D. Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proc. of LREC*, volume 6.

S. Vogel, S. Hewavitharana, M. Kolss, and A. Waibel. 2004. The ISL Statistical Machine Translation System for Spoken Language Translation. In *International Workshop on Spoken Language Translation (IWSLT 2004)*, Kyoto, Japan.