Extending Memory-Based Machine Translation to Phrases

Maarten van Gompel, Antal van den Bosch, and Peter Berck

Tilburg centre for Cognition and Communication
Tilburg University

Abstract

We present a phrase-based extension to memory-based machine translation. This form of example-based machine translation employs lazy-learning classifiers to translate fragments of the source sentence to fragments of the target sentence. Source-side fragments consist of variable-length phrases in a local context of neighboring words, translated by the classifier to a target-language phrase. We compare three methods of phrase extraction, and present a new decoder that reassembles the translated fragments into one final translation. Results show that one of the proposed phrase-extraction methods—the one used in Moses—leads to a translation system that outperforms context-sensitive word-based approaches. The differences, however, are small, arguably because the word-based approaches already capture phrasal context implicitly due to their source-side and target-side context sensitivity.

1 Introduction

Memory-based machine translation (van den Bosch and Berck 2009, van den Bosch et al. 2007, Canisius and van den Bosch 2009) (MBMT for short) is a variant of example-based machine translation. A key characteristic of MBMT is the use of memory-based classifiers (Daelemans et al. 1997, Daelemans et al. 2007) for the translation step. Memory-based classifiers do not only look up stored translation fragment pairs, but are also able to generate translations when the input does not offer an exact match in memory. A parallel corpus serves as the training material. All sentences in this parallel corpus are tokenised and paired up with their counterparts, and between the words of each sentence pair, an alignment is computed. This alignment serves as the basis from which small source-language fragments in their context can be extracted that are subsequently passed to a classifier for training. Whereas prior research in MBMT composed these fragments from single words in context (van den Bosch and Berck 2009, Canisius and van den Bosch 2009), the approach proposed in this study takes a phrase–one or more words—as the focal element of each fragment.

We thus start from a mapping of fragments in the source language to fragments in the target language extracted from a training corpus. Subsequently, memory-based learning is applied to convert these paired fragments into a memory-based classifier (Daelemans et al. 1997). This classifier can then be used to translate new sentences. Given a sentence to translate, we segment it into various phrase-based fragments; for each fragment, a distribution of possible output fragment translations is predicted by the memory-based classifier. As a final step, all translations

Proceedings of the 20th Meeting of Computational Linguistics in the Netherlands
Edited by: Eline Westerhout, Thomas Markus, and Paola Monachesi.

of these fragments are recombined by a new decoder that searches for a globally optimal translation of the given sentence.

The study builds upon previous research on MBMT (van den Bosch and Berck 2009, van den Bosch et al. 2007, Canisius and van den Bosch 2009). The question addressed here is whether a phrase-based approach improves MBMT. An extension to phrases introduces non-trivial issues; one is how to detect phrases in a parallel training corpus. In the study described in this paper, three methods of phrase extraction are tested and compared. Second, there is the issue of choosing a representation of variable-length phrases in the fixed-length feature vector representation assumed by memory-based classification.

In Section 2 we present our approaches to phrase-based MBMT in detail. In Section 3 the results of a comparative series of experiments are presented and discussed. We formulate our conclusions and starting points for future research in Section 4. The system presented in this paper is available as open source software from `http://ilk.uvt.nl/mbmt/pbmbmt`.

2 Phrase-based memory-based machine translation

An MBMT system divides into a training subsystem, producing a translation model, and a translation subsystem. Figure 1 illustrates the setup of the system. A parallel corpus is used for phrase extraction and example generation, i.e. the generation of translations of source fragments to target fragments. These fragments, with as their main constituent an aligned pair of phrases, are stored into a compressed tree structure during the training phase, this in order to facilitate fast retrieval, but as a side effect memory needs are minimized as well. In testing, unseen source-language sentences in a test corpus are also divided into fragments, which the memory-based classifier maps onto a distribution of target-language fragment translations. A decoder then reassembles all translated fragments together into one sentence, searching through and choosing among alternative solutions.

2.1 Example generation

We assume a word alignment between all sentence pairs in the parallel corpus. Figure 2 (left) illustrates such a word-aligned sentence pair, serving as an example throughout this section. On the basis of this, we create example fragment translations that serve as training examples. On the input side, an example consists of a feature vector representing a source-language fragment; on the output side, the example is labeled with a class, representing a fragment of the target sentence aligning to the source fragment. In prior research (van den Bosch and Berck 2009, Canisius and van den Bosch 2009), the feature vector consisted of one focus word, one context word to the left, and one context word to the right; the class was composed of the target-language word aligned to the focus word, and again one context word to the left, and one to the right. Suppose we translate French to English and look at the word est in Figure 2 (left), then the feature vector would be (inconnu, est, condamné), and the class would be (man,is,wrongly). Note that

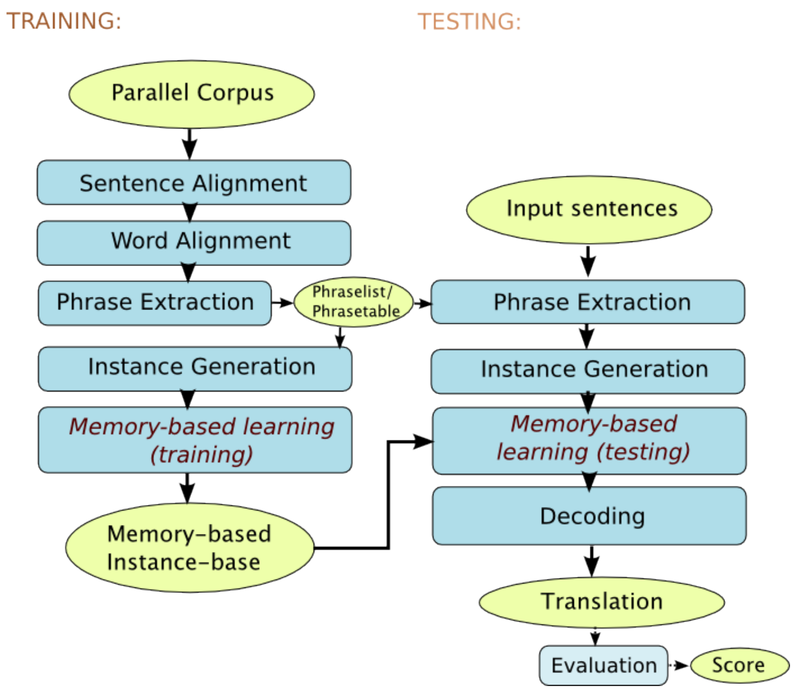

Figure 1: Setup of the phrase-based memory-based machine translation system. The left-hand side corresponds to the training phase; the right-hand side corresponds to the testing phase. Rounded nodes denote data, and square nodes denote processes that manipulate the data.

the class is considered by the classifier as an atomic symbol, but it is decomposed later into its constituents by the decoder. By moving a sliding window over the source sentence, fragments can be generated for all words save for zero-fertility words.

The phrase-based approach we present here is similar. Examples are composed as follows: The feature vector consists of a phrase from the source sentence, with one context word on the left side, and one context word on the right side. The class consists of the target-language phrase that aligns to the source-language phrase, and can optionally also take left and right context words. However, in our study, as detailed below, we found that taking no target-side context produced markedly better results. In representing the source-language side of the example as a feature vector, the variable-width focus phrase can be coded into multiple features. Since phrases can be of arbitrary length and the classifier expects a feature vector of fixed size, this poses a problem. Section 2.3 addresses this issue further.

Suppose that the alignment links the source-language phrase *"l'homme inconnu"* to the target-language phrase *"the unknown man"* in the target language.

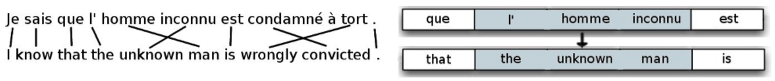
Je sais que l' homme inconnu est condamné à tort .
I know that the unknown man is wrongly convicted .
que
l'
homme
inconnu
est
that
the
unknown
man
is

Figure 2: *Left:* A word alignment between a French and English sentence, *Right:* A phrase-based training example in context

Figure 2 (right) illustrates the phrase-based training example generated for this aligned pair of phrases.

2.2 Phrase extraction

A first task in this study is how to determine phrases in the source- and target-language sentences in the parallel corpus available for training. One solution is to employ the same type of method as used in phrase-based statistical machine translation, making use of a phrase-translation table (Koehn 2004). Such a table lists aligned phrases in both source and target language, assigning conditional probabilities to each. These aligned phrase pairs are computed statistically over the entire parallel corpus by taking the intersection of source-to-target and target-to-source word alignments (Koehn 2004), and this can subsequently be extended by using an algorithm that incrementally adds certain points from the union of the two alignments (Och and Ney 2003). We use the implementation in Moses (Koehn et al. 2007) to this end.

In addition to this first method, we include two other approaches to phrase extraction for comparison. The second method of phrase extraction, henceforth named the *phrase-list approach*, is a straightforward method that extracts frequent n-grams only from the source-language side of the training corpus, and stores this in what we call a *phrase list*. The approach needs a frequency threshold above which an n-gram is included in the phrase list. After exploratory experimentation on test material this threshold was set at 25, but more extensive optimisation can be conducted. Unlike in the phrase-table method, the aligned counterpart of a source-side phrase is computed on the fly. Each source sentence is matched against the phrase list, and whenever a phrase is found, we follow the word-alignments from the phrase and assume that the sequence of words it points to is the aligned target phrase, possibly with intervening fertility words.

Using phrases from either a phrase-translation table or phrase-list, we can never expect to obtain full coverage of test sentences. To decrease problems of low coverage and data sparsity, we defined a phrase to consist of *one or more* words. In addition to phrase extraction, we always generate word-based fragments using the same word-oriented approach as used in prior research. This makes the phrase-based approach an extension of the word-based approach. Given the same parallel corpus and input sentences, the training and test examples in phrase-based MBMT are a superset of those in word-based MBMT.

Due to the phrase-based character of our approach, a word in the source sentence can be part of the focus of a feature vector multiple times. A single word always generates its word-based example, but there may also be one or more extracted phrases that the word is a part of. This occurs in both training and test examples. The latter has an important side effect that has an impact on the decoding process we describe below: if there are multiple examples covering the same words, then there will be multiple possible fragmentations of the input sentence (see also Figure 4).

The third phrase-extraction method is *marker-based chunking*, which segments a sentence into non-overlapping chunks, splitting whenever so-called marker words occur. Marker words are typically defined as closed-class function words, and overlap significantly with the most frequent words in most corpora. Marker-based chunks are generated whenever a new marker word occurs at the beginning of the sentence or after a non-marker (or content) word. Thus, each chunk contains at least one non-marker word. When generating training instances, both the source sentence and the target sentence are chunked independently. Figure 3 shows an example of marker-based chunking, in which arrows point at the markers, each at the head of a chunk. The idea behind marker-based chunking is rooted in the Marker Hypothesis (Green 1979), an idea from psycholinguistics that posits that all languages are marked for surface syntax by a specific closed set of lexemes or morphemes (van den Bosch et al. 2007). Marker-based chunking is a phrase extraction strategy that differs from the previous two in the sense that it does not use word n-gram statistics. It has already been employed in a previous study of MBMT (van den Bosch et al. 2007), inspired in turn by its earlier application in EBMT (Gough and Way 2004, Way and Gough 2005).

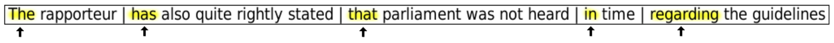
The rapporteur | has also quite rightly stated | that parliament was not heard | in time | regarding the guidelines

Figure 3: Example of marker-based chunking.

The next step is to align the marker-based chunks in the source and target sentences, on the basis of the word alignments already at our disposal. This procedure, described in (van den Bosch et al. 2007), aims to find the target chunk that has the highest probability of being aligned to the source chunk. We effectively estimate $P(C_t|C_s)$ for each source chunk—where C_s is a chunk in the source sentence and C_t a chunk in the target sentence—and align the source chunk with the most probable target chunk. For accurate results the alignment needs to be performed in the other direction as well, estimating $P(C_s|C_t)$ and aligning each target chunk with the most probable source chunk. The intersection of both alignments is then taken as the resulting most likely chunk alignment.

2.3 Representing phrase-based examples

If we encode the focus phrase of the feature vector in terms of its words and their position in the phrase, we end up with feature vectors of different sizes. However, the memory-based classifier demands a fixed number of features in order to compute its similarity function. There are at least three ways to resolve this problem. First, we can consider the entire phrase as one atomic feature. Second, we can reserve a fixed number n of features, and fill those with position-specific words (such as the final word, the prefinal word, etc.), assigning dummy values to unused feature slots. Third, we can assign separate classifiers to different phrase lengths, assigning examples with a particular phrase-length to a separate classifier trained only on examples of this length. In this setup a master process assigns examples to different classifiers and reassembles their output again for the decoder. All three methods of representation have been tried in our research.

2.4 Decoding

Due to the overlapping nature of extractable phrases, and the fact that we may end up with multiple examples covering the same words in the source sentence, we can speak of various possible *fragmentations* of the source sentence S. We define a fragmentation to be a chain in which the fragments are non-overlapping; each fragment covers a certain range of consecutive words of arbitrary length n in the source sentence S, where $1 \leq n \leq |S|$. In addition each fragment is associated with a left context and right context of a length predetermined during example generation. Figure 4 shows three example fragmentations.

Each test example is mapped by the classifier to a distribution of classes, which are the various target-side translations for the fragment with an associated probability score. This probability score is derived from the class distributions produced by the memory-based classifiers by normalizing the class votes. From the perspective of the decoder, the target-side translations with probability scores are referred to as *hypothesis fragments*. Thus, each source-side fragment will be associated with a collection of one or more hypothesis fragments. Figure 4 illustrates the relation between the fragmentation of a sample sentence, the source-side fragments that are extracted from it, and the target-side hypothesis fragments generated from the source-side fragments.

Having gathered all matching fragments for a given source sentence, the task is to search for “good” fragmentations, leading to the most likely translation. The number of fragmentations tends to increase exponentially in the length of the source sentence. Although we do classify all of them with the memory-based classifier, it is infeasible to subsequently search in the space of all possible rearrangements in complete output sentences. Therefore, a local beam search is used to select a number of good fragmentations. The beam size is rather arbitrarily fixed at 20, so at most 20 fragmentations will be returned. The beam search incrementally adds fragments, maximising a score function that sums the normalized scores of the most likely hypothesis fragments predicted for each source-side fragment

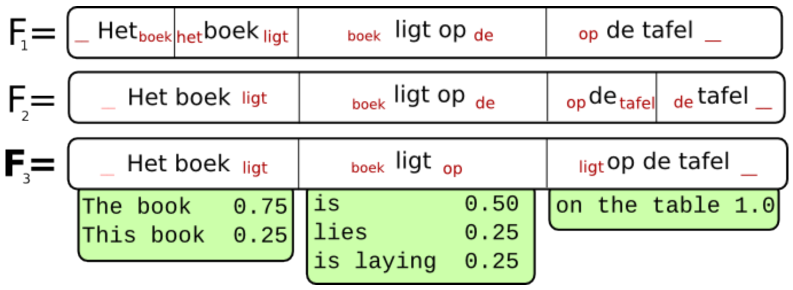

Figure 4: Fragmentations of the sample Dutch input sentence *"Het boek ligt op de tafel"* (The book is on the table). The third fragmentation is expanded to list the target-side hypothesis fragments associated with each of the three source-side fragments the fragmentation is composed of. Context information is printed in small text.

in the current fragmentation. Only fragmentations covering the full sentence are considered as possible solutions.

The fragmentation search described above returns a maximum of 20 fragmentations. For each of these fragmentations of the source sentence, the sentence-global decoding procedure is started. This procedure employs another local beam search to search among alternative translation hypotheses for the given fragmentation. There are thus two search processes going on; one for good fragmentations, and one to search translation hypotheses for each good fragmentation. This makes the phrase-based approach computationally more expensive compared to the word-based approach, as the latter only has one fragmentation of the source sentence.

The actual decoding procedure starts by generating an initial hypothesis: a translation hypothesis in which we simply select for each fragment in the fragmentation the hypothesis fragment with the highest translation probability. We order the hypothesis fragments for the initial hypothesis in the order we find the fragments in the source-sentence fragmentation. The initial hypothesis in Figure 4 thus is *"The book is on the table"*. In this example, the initial hypothesis already happens to generate the best translation, but in most cases there is more searching to do. A hypothesis can be modified in two main ways: (1) the order in which the hypothesis fragments are assembled can be changed, and (2) the choice of hypothesis fragments can be changed, i.e. other hypothesis fragments with an equal or lower translation probability could be tried. To this end, the decoder applies two operations to the initial hypothesis. Each yields new hypotheses, and the best, limited by a beam, are selected. To these hypotheses the operators are applied again. This procedure repeats itself until no better scoring hypotheses can be generated. The first operator is **substitution**. It generates new hypotheses in which a hypothesis fragment of a particular fragment is substituted by another hypothesis fragment from the list. This is done exhaustively within the list of hypothesis fragments (bounded by the first beam search). For each fragment, substitutions are made using all hypothesis fragments that have not undergone a substitution operation in a previous decoding round. The second operator is the **swap** operation, in which we swap the location of two hypothesis fragments. This again is done in an exhaustive fashion such that all possible swaps are made. Each fragment swaps places with all neighboring fragments that are within a certain maximum swap distance, each swap yielding a new hypothesis. An extra parameter specifies the maximum range over which a swap can occur; we set this at two, but exploratory optimisation showed changing this value had little to no effect, the reason is still unknown and is a topic that will be addresses in future research.

In sum, the core decode process, implemented as a local beam search, is called with the initial hypothesis, and then computes all possible substitutions and all possible swaps, resulting in a high number of new hypotheses. This constitutes an informed search problem that seeks to maximise the score of the solution hypotheses. The k best hypotheses are selected, k being the width of the beam, with the constraint that they must be better than the current hypothesis. For each of these k hypotheses the procedure is again repeated, calculating all possible substitutions and swaps, until the point that no better hypotheses can be found. The algorithm can be formalised as in Algorithm 1:

Algorithm 1 Core-Decoder

Require: A set $X_{current}$ containing hypotheses, initially called with only the initial hypothesis as element, and a beam width k.
Ensure: The list containing the k best hypotheses found

```
1: $X_{next} \Leftarrow \emptyset$
2: for all $H_{current} \in X_{current}$ do
3:   for all $H_{next} \in Substitutions(H_{current}) \cup Swaps(H_{current})$ do
4:     if $Score(H_{next}) > Score(H_{current})$ then
5:       $X_{next} \Leftarrow X_{next} \cup \{H_{current}\})$
6:     end if
7:   end for
8: end for
9: if $X_{next} == \emptyset$ then
10:   return $X_{current}$
11: else
12:   $X_{next} \Leftarrow$ Best $k$ hypotheses in $X_{next}$
13:   return $CoreDecoder(X_{next}, k)$
14: end if
```

The success of the decoding algorithm depends on the sentence-level score function it maximises. For each hypothesis, a score is computed that expresses the quality of the hypothesis. A good translation should preferably maximise both *fidelity* and *fluency*. Quantified estimators for these two components are present in the decoder developed for the present study. A quantification of fluency is provided by a trigram-based statistical language model with back-off smoothing and

Good-Turing smoothing on the target language (Stolcke 2002), while fidelity is expressed by the normalized pseudo-probabilities generated by the memory-based translation model. More precisely, the score function for a hypothesis H is made up of the product of the translation probability and the distortion score of the given hypothesis, as shown in Equation 3.1:

$$\text{TranslationModel}(H) = \text{ClassifierScore}(H) \cdot \text{DistScore}(H) \tag{3.1}$$

$$\text{ClassifierScore}(H) = \prod_{i=1}^{|H|} P(classification_i^{weight}) \tag{3.2}$$

$$\text{DistScore}(H) = \prod_{i=1}^{|H|} distortion\_constant^{distance(hfragment_i, hfragment_{i-1})} \tag{3.3}$$

The sentence-level score $\text{ClassifierScore}(H)$ (Equation 3.2) is the product of the translation probabilities of all selected hypothesis fragments that make up the hypothesis. Recall that these translation probabilities are derived from the classifier output, which predicted a distribution of hypotheses fragments as illustrated in Figure 4. The translation probabilities can be given an extra weight parameter by raising them to a certain power.

In addition, a distortion score $\text{DistScore}(H)$ (Equation 3.3) is computed by raising a distortion constant to the power of a measure of distance between two fragments in the original source sentence. The distortion score thus penalises the reordering of fragments; the greater the distance over which an hypothesis fragment is relocated, the lower the distortion score will be. The distortion constant itself is a value between 0 and 1, where the former disallows any reordering of the hypothesis fragments, and the latter does not impose any restrictions at all. This is an admittedly crude factor that may fit the language pair, but will tend to favor ungrammatical and undistorted target sequences over grammatical but reordered target sequences.

One straightforward manner in which to select the final translation would be to select the hypothesis that has the highest score. Yet, this foregoes the fact that among the solutions considered by the decoder, there may be hypotheses representing the same output sentence, even though they are composed of different hypothesis fragments. In the final solution search algorithm we therefore take the sum of the scores of all hypotheses that produce the same output sequence of words in the target language across all of the hypotheses considered.

The last remaining step is to select the target-language sentence for which the sum of the scores of the hypotheses that generated this sentence is maximal. This is the translation generated by the system.

3 Results

Experiments were performed on two parallel corpora, in which we focused only on Dutch to English translation. The first is OpenSubtitles (Tiedemann and Nygaard 2004) consisting of user-contributed subtitles for movies. We generated a training set split consisting of $286,160$ sentence pairs. The second corpus is EMEA (Tiedemann and Nygaard 2004), a medical and largely formulaic text corpus from which we split a training set of $871,180$ sentence pairs. From both corpora we also extracted non-overlapping development and test sets of $1,000$ sentences each.

Several exploratory parameter optimisation experiments were carried out on development data in order to assess the effect of the decoder and some of the parameters. One outcome of these explorations is that omitting target-side context, as used in prior research where target-side fragments constituted trigrams of words (van den Bosch and Berck 2009, Canisius and van den Bosch 2009), greatly improves results. In an experiment on the OpenSubtitles corpus, the BLEU score on development data increased from 0.1211 with target-side context to 0.2184 when target-side context is removed and only target-side phrases are predicted. We attribute this effect mainly to the increase in sparsity of classes when adding context, adding to the sparsity of the phrases themselves.

The main question addressed in this study is whether a phrase-based approach to MBMT (PBMBMT) improves upon the previous word-based approaches (MBMT (van den Bosch and Berck 2009), CSIMT (Canisius and van den Bosch 2009)). In both these methods, the feature vector as well as the class consist of a trigram: one focus word, one left-context word, and one right-context word. CSIMT employs a decoder based on Constraint Satisfaction Inference, and searches through the space of possible target-level translations on the basis of a hill-climbing search (Germann 2003). As our decoder performs a similar search, we compare against this variant. For completeness, we also compare to the MBMT variant described in (van den Bosch and Berck 2009) that maps source trigrams to output trigrams, and uses a decoder that is purely based on target trigram overlap. This overlap-based decoder was introduced in (van den Bosch et al. 2007), where it was shown to outperform a marker-based variant of MBMT.

Other subquestions addressed in this study are: How do the three phrase-extraction methods perform and compare? What example format is best? Indications for answering these questions can be found in Table 3.1. Note that in this table we also list results produced by the PBMBMT decoder when run without using any phrase-extraction method (named wb-PBMBMT), making it operate on a word-based level similar to word-oriented CSIMT, with the notable difference that target-side context is excluded in all PBMBMT experiments. This word-based system offers a baseline for assessing the effectiveness of the phrase-extraction methods, and it can be compared to the word-based decoders reported in earlier work (van den Bosch and Berck 2009, Canisius and van den Bosch 2009).

With respect to the three phrase extraction methods, the Moses (Koehn et al. 2007) phrase-table method performs best overall. The other two methods, especially marker-based chunking, perform below the word-based PBMBMT baseline. This may be attributed to the fact that the phrase-translation table is computed using the two-way alignment statistics of the parallel corpus, whilst the other two methods only rely on source-side statistics, and the aligned counterparts of the phrases are sought in an ad-hoc and per-sentence fashion. The two predecessor

OpenSubtitles

	Extraction	Single/multi	Translation performance metrics				
Decoder	Method	classifier	BLEU	NIST	METEOR	WER	PER
MBMT	-	-	0.1631	4.243	0.3835	68.39	61.33
CSIMT	-	-	0.2002	4.750	0.4431	68.42	55.18
wb-PBMBMT	-	-	0.2163	**5.136**	**0.4644**	55.23	**48.22**
PBMBMT	phrase table	single	**0.2300**	5.055	0.4623	54.47	49.18
PBMBMT	phrase table	multi	0.2256	5.004	0.4583	55.28	49.74
PBMBMT	phrase table	atomic	0.1142	3.026	0.3201	72.81	68.28
PBMBMT	phrase list	single	0.2190	4.980	0.4543	**54.09**	48.77
PBMBMT	phrase list	multi	0.2184	4.975	0.4529	**54.09**	48.79
PBMBMT	marker-based	single	0.1003	2.935	0.3057	76.79	71.16
PBMBMT	marker-based	multi	0.1394	3.360	0.3437	66.40	62.38

EMEA

	Extraction	Single/multi	Translation performance metrics				
Decoder	method	classifier	BLEU	NIST	METEOR	WER	PER
MBMT	-	-	0.2533	5.115	0.4801	72.78	63.66
CSIMT	-	-	0.3013	5.938	0.5333	63.00	**50.85**
wb-PBMBMT	-	-	0.2715	5.600	0.5381	65.99	57.25
PBMBMT	phrase table	single	0.3075	6.011	**0.5455**	59.00	52.02
PBMBMT	phrase table	multi	**0.3078**	**6.019**	0.5449	**58.76**	51.63
PBMBMT	phrase list	single	0.2440	5.352	0.4946	62.74	56.67
PBMBMT	phrase list	multi	0.2440	5.378	0.4967	62.82	56.86
PBMBMT	marker-based	multi	0.2370	4.612	0.4513	74.37	66.78

Table 3.1: Main results on the **OpenSubtitles** and **EMEA** corpora, Dutch to English. Selected parameters were a distortion constant of 0.25, a translation weight of 3, and a maximum swap distance of 5, and a beam width of 1. Note however that in later experiments, slightly higher results have been reported with a beam width of five.

systems, MBMT (van den Bosch and Berck 2009) and CSIMT (Canisius and van den Bosch 2009), are both outperformed by the Moses phrase-table method, and as reported earlier in (Canisius and van den Bosch 2009), the CSIMT method in turn tends to outperform the MBMT method with the trigram overlap-based decoder on all metrics. On the EMEA corpus, CSIMT outperforms the word-based PBMBMT, and performs relatively close to PBMBMT with the Moses phrase-table approach.

We thus observe that the advantage of phrase-based MBMT compared to earlier word-oriented approaches, including the closest comparable system, the word-based PBMBMT baseline (wb-PBMBMT in the table) that restricts itself to words and uses the same decoder as PBMBMT, turns out to be limited. This is a surprising outcome. We may posit that sparsity plays a role here; phrases are by definition less prevalent than single words. The omission of context in classes (in contrast to CSIMT, which maps to trigrams of words) attempts to compensate for this to a certain extent. Another reason for the lack of a clear difference between word-based and phrase-based MBMT may be sought in the fact that even in word-oriented CSIMT there is already a significant but implicit presence of phrasal context. Essentially we are comparing *phrasal context inherent to the phrases themselves* in

PBMBMT, against *phrasal context implicit in the input and output word trigrams* in CSIMT. Often, and most clearly with phrases of three words, the two approaches are mapping about the same input to the same output. The limited gain of the phrase-based approach may stem from the added value of the fact that PBMBMT is not restricted to trigrams, and can vary between whatever is the strongest n-gram, including unigrams. We found that in an experiment on the OpenSubtitles corpus, using the phrase-table method and multi-classifier format, on average 78% of the hypothesis fragments of the predicted translations is in fact a unigram.

Concerning the example format, reserving space for a fixed number of position-specific features (i.e. words) in a single classifier versus distributing different phrase-lengths over multiple classifiers perform more or less on a par. A third format, in which we took a phrase to be an atomic entity, rather than splitting the word over different features, proves to be a poor method of representation, as it yields significantly lower scores (see the entry marked "atomic" in Table 3.1).

In addition to these comparisons, we compared the phrase-based memory-based machine translation approach to state-of-the-art machine translation approaches. Table 3.2 compares PBMBMT with three other systems. The first two, Moses (Koehn et al. 2007) and Google Translate[1], are phrase-based statistical machine translation systems, while Systran is a largely rule-based system. Phrase-based memory-based translation does not approach the performance of the statistical systems. On the other hand, PBMBMT outperforms Systran on all metrics.

No	Corpus	System	BLEU	NIST	METEOR	WER	PER
1	OpenSub	Moses	**0.3289**	**5.903**	**0.5408**	53.29	46.96
2	OpenSub	Google	0.3056	5.790	0.5224	**50.1**	**45.08**
3	OpenSub	PBMBMT	0.2300	5.055	0.4623	54.47	49.18
4	OpenSub	Systran	0.1749	4.583	0.4500	60.77	54.61
1	EMEA	Moses	**0.4701**	**7.059**	**0.6501**	**46.55**	**39.36**
2	EMEA	Google	0.3918	6.377	0.5830	57.57	50.44
3	EMEA	PBMBMT	0.3075	6.011	0.5455	59.00	52.02
4	EMEA	Systran	0.2895	5.472	0.5366	63.24	55.14

Table 3.2: A comparison with state-of-the-art systems

We do not yet have a clear insight into why exactly PBMBMT underperforms in relation to state-of-the-art Phrase-based Statical Machine Translation systems such as as Moses. One cause may be the extensive usage of minimal error training (MERT) on development material, for hyperparameter tuning in Moses, which we have not explored.

[1] `http://translate.google.com`

4 Conclusions and future research

The study described in this paper has demonstrated how memory-based machine translation can be extended from translating fixed-length word trigrams to translating phrases of arbitrary length. We compared three methods of phrase extraction, of which the Moses phrase-translation table approach emerges as the best solution, in fact the only solution scoring above baseline.

Prior research in MBMT such as the recent CSIMT approach (Canisius and van den Bosch 2009) relied partly on target-side context, making use of the overlap between predicted target-side hypothesis fragments (word trigrams) in decoding. The current study shows that ignoring target-side context produces significantly better results in a phrase-based approach, and even performs well in a word-based mode. This can be credited to the decrease in sparsity in the output class space. Moreover, removing this context can be justified by the fact that context becomes less relevant in phrase-based approaches, as target-side phrases capture enough internal context themselves.

Nevertheless, the impact of phrases in comparison to word-based MBMT has been shown to be limited. A potential explanation for this limited effect is that earlier word-based MBMT approaches can be seen as implicitly phrase-based already. The approach followed in (van den Bosch and Berck 2009, Canisius and van den Bosch 2009) maps trigrams of source-side words to trigrams of target-side words, implicitly capturing all phrases up to length three. In this perspective, our current approach changes this only slightly by turning the source-side trigrams into variable-width phrases surrounded by their left and right neighboring words, and predicting variable-width target-side phrases, including single words. In one of our experiments, we found that 78% of fragments in the predicted translation, consists of such single words.

With respect to decoding, we observed in preliminary experiments that starting with an initial hypothesis that follows the order of the source-side fragments as a starting point is a better heuristic than starting with an empty hypothesis and incrementally adding hypothesis fragments. Moreover, subsequently applying substitution and swap operations appears a viable strategy for improving upon this initial hypothesis, even though the resulting gain is modest. Phrase-based decoding differs from word-based decoding in that it needs to take into consideration alternative fragmentations of the input sentence. A strategy has been employed that starts the decoding process with a beam-constrained number of best fragmentations (where goodness is estimated from the pseudo-probabilities generated by the memory-based classifier), and in the end combines the results to select the highest scoring candidate. There are thus essentially two inter-dependent search problems to be solved. Despite the fact that this increases the complexity of decoding, the strategy of applying substitutions and swaps in a hill-climbing search until convergence, results in performing the decoding task in a limited time-span.

Besides hyperparameter optimisation of both the memory-based classifier and the decoder through minimal error training, other options for future research are testing the system on different, more distantly related, language pairs, and the inclusion of richer (e.g. linguistic) features such as part-of-speech tags and lemmas. We believe that considerable improvement can be obtained by improving the decoder. In future work it could be extended with more operations, such as a delete operation powered by a null model; moreover, an alternative should be sought for its current crude distortion factor. The findings with regard to omission of target-side context could be tested and incorporated into the strategy proposed in CSIMT (Canisius and van den Bosch 2009).

Acknowledgement

This research was funded by NWO, the Netherlands Organisation for Scientific Research, as part of the Vici project "Implicit Linguistics".

References

Canisius, S. and A. van den Bosch (2009), A constraint satisfaction approach to machine translation, *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT-2009)*, pp. 182–189.

Daelemans, W., A. van den Bosch, and T. Weijters (1997), Igtree: Using trees for compression and classification in lazy learning algorithms., *Artificial Intelligence Review* **11** (1-5), pp. 407–423. http://dblp.uni-trier.de/db/journals/air/air11.html#DaelemansBW97.

Daelemans, W., J. Zavrel, K. van der Sloot, and A. van den Bosch (2007), Timbl: Tilburg memory based learner, version 6.1, reference guide, *Technical Report ILK-07-07*, Tilburg University, Tilburg, The Netherlands.

Germann, U. (2003), Greedy decoding for statistical machine translation in almost linear time, *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 1–8.

Gough, N. and A. Way (2004), Robust large-scale EBMT with marker-based segmentation, *Proceedings of the Tenth Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2004)*, Baltimore, Maryland, pp. 95–104.

Green, T. (1979), The necessity of syntax markers. two experiments with artificial languages, *Journal of Verbal Learning and Behavior* **18**, pp. 481–496.

Koehn, P. (2004), Pharaoh: A beam search decoder for phrase-based statistical machine translation models., *in* Frederking, R.E. and K. Taylor, editors, *Proceedings of the American Machine Translation Association*, Vol. 3265 of *Lecture Notes in Computer Science*, Springer, pp. 115–124.

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., *ACL*, The Association for Computer Linguistics.

Och, F.J. and H. Ney (2003), A systematic comparison of various statistical alignment models., *Computational Linguistics* **29** (1), pp. 19–51. http://dblp.uni-trier.de/db/journals/coling/coling29.html#OchN03.

Stolcke, A. (2002), Srilm – an extensible language modeling toolkit. http://citeseer.ist.psu.edu/stolcke02srilm.html.

Tiedemann, J. and L. Nygaard (2004), The OPUS corpus - parallel and free, *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC04)*, pp. 26–28.

van den Bosch, A. and P. Berck (2009), Memory-based machine translation and language modeling, *The Prague Bulletin of Mathematical Linguistics* **91**, pp. 17–26.

van den Bosch, A., N. Stroppa, and A. Way (2007), A memory-based classification approach to marker-based EBMT, *Proceedings of the METIS-II Workshop on New Approaches to Machine Translation*, pp. 63–72.

Way, A. and N. Gough (2005), Comparing example-based and statistical machine translation, *Natural Language Engineering* **11** (3), pp. 295–309.