Combining EBMT, SMT, TM and IR Technologies for Quality and Scale

Sandipan Dandapat[1], **Sara Morrissey**[1], **Andy Way**[2], **Joseph van Genabith**[1]
[1] CNGL, School of Computing
Dublin City University, Glasnevin, Dublin 9, Ireland
{sdandapat,smorri,josef}@computing.dcu.ie
[2] Applied Language Solutions, Delph, UK
andy.way@appliedlanguage.com

Abstract

In this paper we present a hybrid statistical machine translation (SMT)-example-based MT (EBMT) system that shows significant improvement over both SMT and EBMT baseline systems. First we present a runtime EBMT system using a subsentential translation memory (TM). The EBMT system is further combined with an SMT system for effective hybridization of the pair of systems. The hybrid system shows significant improvement in translation quality (0.82 and 2.75 absolute BLEU points) for two different language pairs (English–Turkish (En–Tr) and English–French (En–Fr)) over the baseline SMT system. However, the EBMT approach suffers from significant time complexity issues for a runtime approach. We explore two methods to make the system scalable at runtime. First, we use an heuristic-based approach. Secondly, we use an IR-based indexing technique to speed up the time-consuming matching procedure of the EBMT system. The index-based matching procedure substantially improves run-time speed without affecting translation quality.

1 Introduction

State-of-the-art phrase-based SMT (Koehn, 2010a) is the most successful MT approach in many large scale evaluations, such as WMT,[1] IWSLT[2] etc. At the same time, work continues in the area of EBMT. Some recent EBMT systems include Cunei (Phillips, 2011), CMU-EBMT (Brown, 2011) and OpenMaTrEx (Dandapat et al., 2010). The success of an SMT system often depends on the amount of parallel training corpora available for the particular language pair. However, low translation accuracy has been observed for language pairs with limited training resources (Islam et al., 2010; Khalilov et al., 2010). SMT systems effectively discard the actual training data once the models (translation model and language model) have been estimated. This can lead to their inability to guarantee good quality translation for sentences closely matching those in the training corpora. By contrast, EBMT systems usually maintain a linked relationship between the full sentence pairs in source and target texts. Because of this EBMT systems can often capture long range dependencies and rich morphology at runtime. In contrast to SMT, however, most EBMT models lack a well-formed probability model, which restricts the use of statistical information in the translation process.

Keeping these in mind, our objective is to develop a good quality MT system choosing the best approach for each input in the form of a hybrid SMT-EBMT approach. It is often the case that an EBMT system produces a good translation where SMT systems fail and vice versa (Dandapat et al., 2011).

An EBMT system relies on past translations to derive the target output for a given input. Runtime EBMT approaches generally do not include any training stage, which has the advantage of not having to depend on time-consuming preprocessing. On the other hand, their runtime complexity can be considerable. This is due to the time-consuming matching stage at runtime that finds the example

[1] http://www.statmt.org/wmt11/
[2] http://www.iwslt2011.org/

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 48–58,
Avignon, France, April 23 - 27 2012.

(or set of examples) which most closely matches the source-language sentence to be translated. This matching step often uses some variation of string edit-distance measures (Levenshtein, 1965) which has quadratic time complexity.[3] This is quite time-consuming even when a moderate amount of training examples are used for the matching procedure.

We adopt two alternative approaches to tackle the above problem. First we use heuristics which are often useful to avoid some of the computations. For a input sentence, in the matching process, we may not need to compute the string edit distance with all sentences in the example base. In order to prune some of the computation, we rely on the fact that the input sentence and its closest match sentence from the example-base are likely to have a similar sentence length. Search engine indexing is an effective way of storing data for fast and accurate retrieval of information. During retrieval, a set of documents are extracted based on their similarity to the input query. In our second approach, we use this concept to efficiently retrieve a potential set of suitable candidate sentences from the example-base to find the closest match. We index the entire example-base considering each source-side sentence as a document for the indexer. We show that improvements can be made with our approach in terms of time complexity without affecting the translation quality.

The remainder of this paper is organized as follows. The next section presents work related to our EBMT approach. Section 3 describes the MT systems used in our experiments. Section 4 focuses on the two techniques used to make the system scalable. Section 5 presents the experiments in detail. Section 6 presents and discusses the results and provides an error analysis. We conclude in Section 7.

2 Related Work

The EBMT framework was first introduced by Nagao (1984) as the "MT by analogy principle". The two main approaches to EBMT are distinguished by the inclusion or exclusion of a preprocessing/training stage. Approaches that incorporate a training stage are commonly called "compiled approaches" (Cicekli and Güvenir, 2001). Approaches that do not include a training stage are often referred to as "pure" or "runtime" EBMT approaches, e.g. (Lepage and Denoual, 2005). These approaches have the advantage that they do not depend on any time-consuming preprocessing stages. On the other hand, their runtime complexity can be considerable.

EBMT is often linked with the related concept of *translation memory* (TM). A TM essentially stores source- and target-language translation pairs for effective reuse of previous translations originally created by human translators. TMs are often used to store examples for EBMT systems. After retrieving a set of examples with associated translations, EBMT systems automatically extract translations of suitable fragments and combine them to produce a grammatical target output.

Phrase-based SMT systems (Koehn, 2010a), produce a source–target aligned subsentential phrase table which can be adapted as an additional TM to be used in a CAT environment (Simard, 2003; Biçici and Dymetman, 2008; Bourdaillet et al., 2009; Simard and Isabelle, 2009). Koehn and Senellart (2010b) use SMT to produce the translation of the non-matched fragments after obtaining the TM-based match. EBMT phrases have also been used to populate the knowledge database of an SMT system (Groves et al., 2006). However, to the best of our knowledge, the use of SMT phrase tables within an EBMT system as an additional sub-sentential TM has not been attempted so far. Some work has been carried out to integrate MT in a CAT environment to translate the whole segment using the MT system when no sufficiently well matching translation unit (TU) is found in the TM. The TransType system (Langlais et al., 2002) integrates an SMT system within a text editor to suggest possible continuations of the translations being typed by the translator. By contrast, our approach attempts to integrate the subsentential TM obtained using SMT techniques within an EBMT system.

3 MT Systems

The SMT system used in our hybrid SMT-EBMT approach is the vanilla Moses[4] decoder.

[3] Ukkonen (1983) gave an algorithm for computing edit-distance with the worst case complexity $O(md)$, where m is the length of the string and d is their edit distance. This is effective when $m \gg d$. We use word-based edit distance, so m is shorter in length.

[4] http://www.statmt.org/moses/

Moses (Koehn et al., 2007) is a set of SMT tools that include routines to automatically train a translation model for any language pair and an efficient decoder to find the most probable translation. Due to lack of space and the wide usage of Moses, here we focus more on the novel EBMT system we have developed for our hybrid SMT-EBMT approach. The EBMT system described in this section is based on previous work (Dandapat et al., 2010) and some of the material has been reproduced here to make the paper complete.

Like all other EBMT systems, our particular approach comprises three stages: matching, alignment and recombination. Our EBMT system also uses a subsentential TM in addition to the sentence aligned example-base. Using the original TM as a training set, additional subsentential TUs (words and phrases) are extracted from it based on word alignments and phrase pairs produced by Moses. These subsentential TUs are used for alignment and recombination stages of our EBMT system.

3.1 Building a Subsentential TM for EBMT

A TM for EBMT usually contains TUs linked at the sentence, phrasal and word level. TUs can be derived manually or automatically (e.g. using the marker-hypothesis (Groves et al., 2006)). Usually, TUs are linguistically motivated translation units. In this paper however, we explore a different route, as manual construction of high-quality TMs is time consuming and expensive. Furthermore, only considering linguistically motivated TUs may limit the matching potential of a TM. Because of this, we used SMT technology to automatically create the subsentential part of our TM at the phrase (i.e. no longer necessarily linguistically motivated) and word level. Based on Moses word alignment (using GIZA++ (Och and Ney, 2003)) and phrase table construction, we construct the additional TM for further use within an EBMT approach.

Firstly, we add entries to the TM based on the aligned phrase pairs from the Moses phrase table using the following two scores:

1. Direct phrase translation probabilities: $\phi(t|s)$
2. Direct lexical weight: $lex(t|s)$

Table 1 shows an example of phrase pairs with the associated probabilities learned by Moses. We keep all target equivalents in a sorted order based on the above probabilities. This helps us in the matching procedure, but during recombination we only consider the most probable target equivalent. The following shows the resulting TUs in the TM for the English source phrase *a hotel*.

Table 1: Moses phrase equivalence probabilities.

English (s)	**Turkish (t)**	$p(t\|s)$	$lex(t\|s)$
a hotel	bir otel	0.826087	0.12843
a hotel	bir otelde	0.086957	0.07313
a hotel	otel mi	0.043478	0.00662
a hotel	otel	0.043478	0.22360

a hotel $\Leftrightarrow$ {*bir otel*, *bir otelde*, *otel*, *otem mi*}

Secondly, we add entries to the TM based on the source-to-target word-aligned file. We also keep the multiple target equivalents for a source word in a sorted order. This essentially adds source- and target-language equivalent word pairs into the TM. Note that the entries in the TM may contain incorrect source-target equivalents due to unreliable word/phrase alignments produced by Moses.

3.2 EBMT Engine

The overview of the three stages of the EBMT engine is given below:

Matching: In this stage, we find a sentence pair $\langle s_c, t_c \rangle$ from the example-base that closely matches with the input sentence s. We used a fuzzy-match score (FMS) based on a word-level edit distance metric (Wagner and Fischer, 1974) to find the closest matching source-side sentence from the example-base ($\{s_i\}_1^N$) based on Equation (i).

$$\text{score}(s, s_i) = 1 - \text{ED}(s, s_i)/\max(|s|, |s_i|) \quad (i)$$

where $|x|$ denotes the length (in words) of a sentence, and $\text{ED}(x, y)$ refers to the word-level edit distance between x and y. The EBMT system considers the associated translation t_c of the closest matching source sentence s_c, to build a skeleton for the translation of the input sentence s.

Alignment: After retrieving the closest fuzzy-matched sentence pair $\langle s_c, t_c \rangle$, we identify the non-matching fragments from the skeleton translation t_c in two steps.

Firstly, we find the matched and non-matched segments between s and s_c using edit distance trace. Given the two sentences (s and s_c), the algorithm finds the minimum possible number of operations (substitutions, additions and deletions) required to change the closest match s_c into the input sentence s. For example, consider the input sentence $s = w_1\, w_2\, w_3\, w_4\, w_5\, w_6\, w_7\, w_8$ and $s_c = w'_1\, w'_3\, w_4\, w_5\, w_7\, w_8\, w'_9$. Figure 1 shows the matched and non-matched sequence between s and s_c using edit-distance trace.

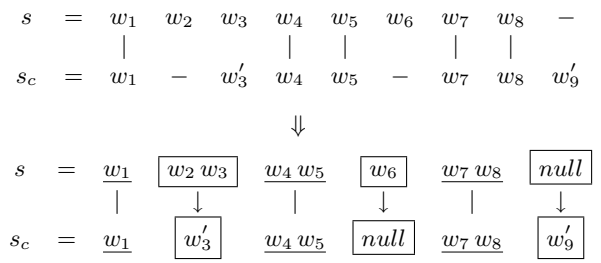


Figure 1: Extraction of matched (underlined) and non-matched (boxed) segments between s and s_c.

Secondly, we align each non-matched segment in s_c with its associated translation using the TM and the GIZA++ alignment. Based on the source-target aligned pair in the TM, we mark the mismatched segment in t_c. We find the longest possible segment from the non-matched segment in s_c that has a matching target equivalent in t_c based on the source-target equivalents in the TM. We continue the process recursively until no further segments of the non-matched segment in s_c can be matched with t_c using the TM. Remaining non-matching segments in s_c are then aligned with segments in t_c using the GIZA++ word alignment information.

Recombination: In the recombination stage, we add or substitute segments from the input sentence s with the skeleton translation equivalent t_c. We also delete some segments from t_c that have no correspondence in s. After obtaining the source segments (needs to be added or substituted in t_c) from the input s, we use our subsentential TM to translate these segments. Details of the recombination process are given in Algorithm 1.

3.3 An Illustrative Example

As a running example, for the input sentence in (1a) the corresponding closest fuzzy-matched sentence pair $\langle s_c, t_c \rangle$ is shown in (1b) and (1c). The portion marked with angled brackets in (1c) are aligned with the mismatched portion in (1b). The character and the following number in angled brackets indicate the edit operation ('s' indicates substitution) and the index of the mismatched segment from the alignment process respectively.

Algorithm 1 recombination(X, TM)

In: source segment X,
subsentential translation memory TM
Out: translation of source segment X

1: mark all words of X as untranslated (untranslatedPortions$(X) \leftarrow \{X\}$)
2: **repeat**
3: $U =$ untranslatedPortions(X)
4: $x =$ longest subsegment in untranslatedPortions(X) such that $(x, t_x) \in$ TM;
5: substitute$(X, x \rightarrow t_x)$ {substitute x with its target equivalent t_x in X}
6: remove x from untranslatedPortions(X)
7: **until** (untranslatedPortions$(X) = U$)
8: return X

1. (a) s: i 'd like a <s#0:*present*> for <s#1:*my mother*> .
   (b) s_c: i 'd like a <s#0:*shampoo*> for <s#1:*greasy hair*> .
   (c) t_c: <s#1:*yağlı saçlar*> için bir <s#0:*şampuan*> istiyorum .

During recombination, we need to replace two segments in (1c) {*yağlı saçlar* (greasy hair) and *şampuan* (shampoo)} with the two corresponding source segments in (1a) {*my mother* and *present*} as an intermediate stage (2) along the way towards producing a target equivalent.

(2)<1:*my mother*> için bir <0:*present*> istiyorum .

Furthermore, replacing the untranslated segments in (2) with the translations obtained using TM, we derive the output translation in (3) of the original input sentence in (1).

(3) <*annem*> için bir <*hediye*> istiyorum .

4 Scalability

The main motivation of scalability is to improve the speed of the EBMT system when using a large example-base. The matching procedure in an EBMT system finds the example (or a set of examples) which closely matches the source-language string to

be translated. All matching processes necessarily involve a distance or similarity measure. The most widely used distance measure in EBMT matching is Levenshtein distance (Levenshtein, 1965; Wagner and Fischer, 1974) which has quadratic time complexity. In our EBMT system, we find the closest sentence at runtime from the whole example-base for a given input sentence using the edit distance matching score. Thus, the matching step of the EBMT system is a time-consuming process with a runtime complexity of $O(nm^2)$, where n denotes the size of the example-base and m denotes the average length (in words) of a sentence. Due to a significant runtime complexity, the EBMT system can only handle a moderate size example-base in the matching stage. However, it is important to handle a large example-base to improve the quality of an MT system. In order to make the system scalable with a larger example-base, we adopt two approaches for finding the closest matching sentences efficiently.

4.1 Grouping

Our first attempt is heuristic-based. We divide the example-base into bins based on sentence length. It is anticipated that the sentence from the example-base that most closely matches an input sentence will fall into the group which has comparable length to the length of the input sentence. First, we divide the example-base E into different bins based on their word-level length $E = \bigcup_{i=1}^{l} E_i$ and $E_i \bigcap E_j = \emptyset$ for all $i \neq j$ where $0 \leq i, j \leq l$. E_i denotes the set of sentences with length i and l is the maximum length of a sentence in E. In order to find the closest match for a test sentence (s of length k), we only consider examples $E_G = \bigcup_{m=0}^{x} E_{k \pm m}$, where x indicates the window size. In our experiment, we consider the value of x from 0 to 2. We find the closest-match s_c from E_G for a given test sentence s. E_G has fewer sentences compared to E which will effectively reduce the time of the matching procedure.

4.2 Indexing

Our second approach to addressing time complexity is to use indexing. We index the complete example-base using an open-source IR engine SMART[5] and retrieve a potential set of candidate sentences (likely to contain the closest match sentence) from the example-base. Unigrams extracted from the sentences of the example-base are indexed using the language model (LM) and complete sentences are considered as retrievable units. In LM-based retrieval we assume that a given query is generated from a unigram document language model. The application of the LM retrieval model in our case returns a sorted list of sentences from the example-base ordered by the estimated probabilities of generating the given input sentence.

In order to improve the run-time performance, we integrate the SMART retrieval engine within the matching procedure of our EBMT system. The retrieval engine estimates a potential set of candidate close-matching sentences from the example-base E for a test sentence s. We assume that the closest source-side match s_c of the input sentence s can take the value from the set $\mathrm{E}_{\mathrm{IR}}(s)$, where $\mathrm{E}_{\mathrm{IR}}(s)$ is the potential set of close-matching sentences computed by the LM-based retrieval engine. We have used the top 50 candidate sentences from $\mathrm{E}_{\mathrm{IR}}(s)$. Since the IR engine tries to retrieve the document (sentences from E) for a given query (input) sentence, it is likely to retrieve the closest match sentence s_c in the set $\mathrm{E}_{\mathrm{IR}}(s)$. Due to a much reduced set of possibilities, this approach improves the run-time performance of the EBMT system without hampering system accuracy. Finding this potential set of candidate sentences will be much faster than traditional edit-distance-based retrieval on the full example-base as the worst case run time of the retriever is $O(\sum_{\forall w_i} s_i)$, where w_i is a word in the input sentence and s_i is the number of sentences in the example-base that contain w_i. Finding a set of candidate sentences took only 0.3 seconds and 116 seconds, respectively, for 414 and 10,000 example input sentences given 20k and 250k sentence example-base in our En–Tr and En–Fr experiment on a 3GHz Core 2 Duo machine with 4GB RAM.

5 Experiments

We conduct different experiments to report the accuracy of our EBMT systems for En–Tr and En–Fr translation tasks. In order to compare the performance of our approaches we use two baseline systems. We use the Moses **SMT** system as one base-

[5] An open source IR system from Cornell University. `ftp://ftp.cs.cornell.edu/pub/smart/`

line. Furthermore, based on the matching step (Section 3.2) of the EBMT approach, we obtain the closest target-side equivalent (the skeleton sentence) and consider this as the baseline output for the input to be translated. This is referred to as **TM** in the experiment below. We will consider this as the baseline accuracy for our EBMT using TM approach.

In addition, we conduct two experiments with our EBMT system. After obtaining the skeleton translation through the matching and alignment steps, in the *recombination* step, we use TM to translate any unmatched segments based on Algorithm 1. We call this $\mathbf{EBMT_{TM}}$.

We found that there are cases where the $\mathrm{EBMT_{TM}}$ system produces the correct translation but SMT fails and vice-versa (Dandapat et al., 2011). In order to further improve translation quality, we use a combination of EBMT and SMT. Here we use some features to decide whether to rely on the output produced by the $\mathrm{EBMT_{TM}}$ system. These features include *fuzzy match score***FMS** (as in (i)) and the number of mismatched segments in each of s, s_c, t_c (**EqUS**[6] as in (1)). We assume that the translations of an input sentence s produced by $\mathrm{EBMT_{TM}}$ and SMT systems are respectively $\mathrm{T_{EBMT}}(s)$ and $\mathrm{T_{SMT}}(s)$. If the value of FMS is greater than some threshold and EqUS exists between s and s_c, we rely on the output $\mathrm{T_{EBMT}}(s)$; otherwise we take the output from $\mathrm{T_{SMT}}(s)$. We refer to this system as $\mathbf{EBMT_{TM} + SMT}$.

To test the scalability of the system, we conducted two more experiments based on the approach described in Section 4. First, we conducted an experiment based on the sentence length-based grouping heuristics (Section 4.1). We refer to this system as $\mathbf{EBMT_{TM} + SMT} + \mathit{group}_i$, where i indicates the window size while comparing the length of the input sentence with the bins. We conduct a second experiment based on the LM-based indexing technique (Section 4.2) we have used to retrieve a potential set of candidate sentences from the indexed example-base. We call this system $\mathbf{EBMT_{TM} + SMT} + \mathit{index}$. Note that the $\mathrm{EBMT_{TM}} + \mathrm{SMT}$ system is used as the baseline accuracy while conducting the experiments for scalability of the EBMT system.

[6] If s, s_c and t_c agree in the number of mismatched segments, EqUS evaluates to 1, otherwise 0.

5.1 Data Used for Experiments

We used two data sets for all our experiments representing two language pairs of different size and type. In the first data-set, we have used the En–Tr corpus from IWSLT09.[7] The training data consists of 19,972 parallel sentences. We used the IWSLT09 development set as our testset which consists of 414 sentences. The IWSLT09 data set is comprised of short sentences (with an average of 9.5 words per sentence) from a particular domain (the C-STAR project's Basic Travel Expression Corpus).

Our second data set consists of an En–Fr corpus from the European Medicines Agency (EMEA)[8] (Tiedemann and Nygaard, 2009). The training data consists of 250,806 unique parallel sentences.[9] As a testset we use a set of 10,000 randomly drawn sentences disjoint from the training corpus. This data also represents a particular domain (medicine) but with longer sentence lengths (with an average of 18.8 words per sentence) compared to the IWSLT09 data.

6 Results and Observations

We used BLEU (Papineni et al., 2002) for automatic evaluation of our EBMT systems. Table 2 shows the accuracy obtained for both En–Tr and En–Fr by the $\mathrm{EBMT_{TM}}$ system described in Section 3. Here we have two baseline systems (SMT and TM) as described in the first two experiments in Section 5.

Table 2: Baseline BLEU scores of the two systems and the scores for $\mathrm{EBMT_{TM}}$ system.

System	Language pairs	
	En–Tr	En–Fr
SMT	23.59	55.04
TM	15.60	40.23
$\mathrm{EBMT_{TM}}$	20.08	48.31

Table 2 shows that $\mathrm{EBMT_{TM}}$ has a lower system accuracy than SMT for both the language pairs, but

[7] http://mastarpj.nict.go.jp/IWSLT2009/2009/12/downloads.html

[8] `http://opus.lingfil.uu.se/EMEA.php`

[9] A large number of duplicate sentences exists in the original corpus (approximately 1M sentences). We remove duplicates and consider sentences with unique translation equivalents.

better scores than TM alone. Tables 3 and 4 show that combining EBMT with SMT systems shows improvements of 0.82 and 2.75 BLEU absolute over the SMT baseline (Table 2) for both the En–Tr and the En–Fr data sets. In each case, the improvement of $\mathrm{EBMT_{TM}} + \mathrm{SMT}$ over the baseline SMT is statistically significant (reliability of 98%) using bootstrap resampling (Koehn, 2004).

Table 3: En–Tr MT system accuracies of the combined systems ($\mathrm{EBMT_{TM}} + \mathrm{SMT}$) with different combining factors. The second column indicates the number (and percentage) of sentences translated by the $\mathrm{EBMT_{TM}}$ system during combination.

System: $\mathrm{EBMT_{TM}} + \mathrm{SMT}$		
Condition	**times $\mathrm{EBMT_{TM}}$ used**	**BLEU (in %)**
FMS>0.85	35 (8.5%)	24.22
FMS>0.80	114 (27.5%)	23.99
FMS>0.70	197 (47.6%)	22.74
FMS>0.80 OR (FMS>0.70 & EqUS)	165 (40.0%)	23.87
FMS>0.85 & EqUS	24 (5.8%)	**24.41**
FMS>0.80 & EqUS	76 (18.4%)	24.19
FMS>0.70 & EqUS	127 (30.7%)	24.08

Table 4: En–Fr MT system accuracies for the combined systems ($\mathrm{EBMT_{TM}} + \mathrm{SMT}$) with different combining factors.

System: $\mathrm{EBMT_{TM}} + \mathrm{SMT}$		
Condition	**times $\mathrm{EBMT_{TM}}$ used**	**BLEU (in %)**
FMS>0.85	3323 (33.2%)	**57.79**
FMS>0.80	4300 (43.0%)	57.55
FMS>0.70	5283 (52.8%)	57.05
FMS>0.60	6148 (61.5%)	56.25
FMS>0.80 OR (FMS>0.70 & EqUS)	4707 (47.1%)	57.46
FMS>0.85 & EqUS	2358 (23.6%)	57.24
FMS>0.80 & EqUS	2953 (29.5%)	57.16
FMS>0.70 & EqUS	3360 (33.6%)	57.08

A particular objective of our work is to scale the runtime EBMT system to a larger amount of training examples. We experiment with the two approaches described in Section 4 to improve the run time of the system. Table 5 compares the run time of the three systems ($\mathrm{EBMT_{TM}}$, $\mathrm{EBMT_{TM}} + \mathit{group}_i$ and $\mathrm{EBMT_{TM}} + \mathit{index}$) for both En–Tr and En–Fr translation. Note that the SMT decoder takes 140 seconds and 310 minutes respectively for En–Tr and En–Fr translation test sets.

Table 5: Running time of the three different systems.

System	Language pairs	
	En–Tr (*seconds*)	En–Fr (*minutes*)
SMT	140.0	310.0
$\mathrm{EBMT_{TM}}$	295.9	2267.0
$\mathrm{EBMT_{TM}} + \mathit{group}_0$	34.0	63.4
$\mathrm{EBMT_{TM}} + \mathit{group}_1$	96.2	183.5
$\mathrm{EBMT_{TM}} + \mathit{group}_2$	148.5	301.4
$\mathrm{EBMT_{TM}} + \mathit{index}$	**2.7**	**2.6**

Both the grouping and indexing methodologies proved successful for system scalability with a maximum speedup of almost 2 orders of magnitude. We also need to estimate the accuracy while combining grouping and indexing techniques with the baseline system ($\mathrm{EBMT_{TM}} + \mathrm{SMT}$) to understand their relative performance. Table 6 provides the system accuracy using the grouping and indexing techniques for both the language pairs. We report the translation quality under three conditions. Similar trends have been observed for other conditions.

6.1 Observations and Discussions

We find that the $\mathrm{EBMT_{TM}}$ system has a lower accuracy on its own compared to baseline SMT for both the language pairs (Table 2). Nevertheless, there are sentences which are better translated by the $\mathrm{EBMT_{TM}}$ approach compared to SMT, although the overall document translation score is higher with SMT. Thus, we combined the two systems based on different features and found that the combined system performs better. The highest relative improvements in BLEU score are 3.47% and 1.05% respectively for En–Tr and En–Fr translation. We found that if an input has a high fuzzy match score (FMS) with the example-base, then the $\mathrm{EBMT_{TM}}$ system does better compared to SMT. With our current experimental setup, we found that an FMS over 0.8 showed an improvement for En–Tr and a FMS over 0.6 showed improvement for En–Fr over the SMT system. Figure 2 shows the effect in the translation

Table 6: BLEU scores of the three different systems for En–Tr and En–Fr under different conditions. i denotes the number of bins considered during grouping.

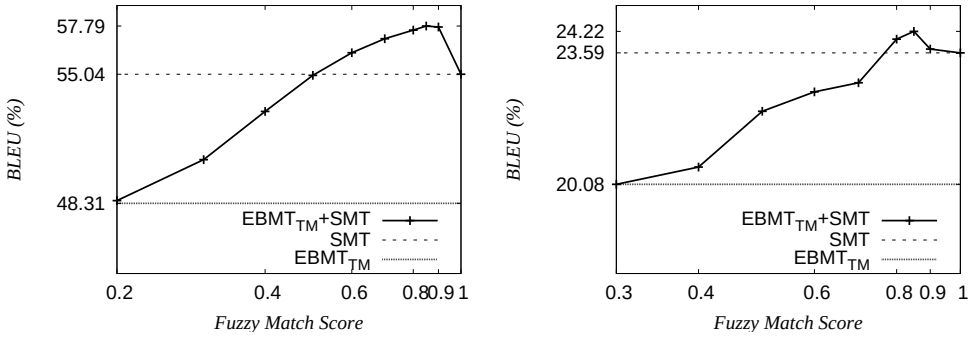
Condition	System				
	$\text{EBMT}_{\text{TM}} + \text{SMT}$	$\text{EBMT}_{\text{TM}} + \text{SMT}$ $+group_i$			$\text{EBMT}_{\text{TM}} + \text{SMT}$ $+index$
		i=0	i=$\pm$1	i=$\pm$2	
En–Tr					
FMS>0.85	24.22	24.18	24.18	24.23	24.24
FMS>0.80 OR (FMS>0.70 & EqUS)	23.87	23.34	23.90	24.40	24.37
FMS>0.85 & EqUS	24.41	24.17	24.38	24.34	24.39
En–Fr					
FMS>0.85	57.79	56.47	57.48	57.76	57.92
FMS>0.80 OR (FMS>0.70 & EqUS)	57.46	55.69	57.07	57.33	57.56
FMS>0.85 & EqUS	57.24	56.48	57.23	57.29	57.32

quality when different FMS thresholds were used to combine the two systems.

However, FMS might not be the only factor for triggering the EBMT_{TM} system. We considered EqUs as another factor which showed improvement for En–Tr but showed negative effect for En–Fr. Though an FMS over 0.7 for En–Tr shows no improvement in overall system accuracy, inclusion of the EqUs feature along with FMS shows improvement. Thus, the EBMT_{TM} system is sometimes more effective when the number of unmatched segment matches in s, s_c and t_c.

These observations show the effective use of our EBMT approach in terms of translation quality. However, we found that the EBMT_{TM} system has a very considerable runtime complexity. In order to translate 414 test sentences from English into Turkish, the basic EBMT system takes 295.9 seconds. The situation becomes worse when using the large example-base for En–Fr translation. Here, we found that the system takes around 38 hours to translate 10k source English sentences into French. This is a significant time complexity by any standard for a runtime approach. However, both grouping and indexing reduce the time complexity of the approach considerably. The time reduction with grouping depends on the number of bins considered to find the closest sentence during the matching stage. Systems with a lower number of bins take less time but cause more of a drop in translation quality. The effect is more prominent with the En–Fr system which uses a larger example-base. We found a drop of absolute 1.32 BLEU points while considering a single bucket whose length is equal to the length of the test sentence. This configuration takes 63 minutes to translate 10k English sentences into French. There is only a drop of 0.03 BLEU points when considering the 5 nearest bins ($\pm$2) for a given test sentence. Nevertheless, there is not much of a reduction but it increases the run time to 5 hours for the translation of 10k sentences. Thus, the group-based method is not effective enough to balance system accuracy and run time.

Incorporation of the indexing technique into the matching stage of EBMT shows the highest efficiency gains in run time. Translating 10k sentences from English into French takes only 158 seconds. It is also interesting to note that with indexing, the BLEU score remained the same or even increased. This is due to the fact that, compared to FMS-based matching, a different closest-matching sentence s_c is selected for some of the input sentences while using indexing, thus resulting in a different outcome to the system. Figure 3 compares the number of times the $\text{EBMT}_{\text{TM}} + \text{SMT} +$ *index* system is used in the hybrid system and the number of same closest-matching sentences selected by $\text{EBMT}_{\text{TM}} + \text{SMT} +$ *index* systems under different conditions for En–Tr. The use of index-based candidate selection for EBMT matching shows effective



(a) En—Fr (b) En—Tr

Figure 2: Effect of FMS in the combined $\mathrm{EBMT_{TM}} + \mathrm{SMT}$ system.

Table 7: The effect of indexing in selection s_c and in final translation.

Input:	zeffix belongs to a group of medicines called antivirals.
Ref:	**zeffix appartient à une classe de médicaments appelés antiviraux.**
baseline $\mathrm{EBMT_{TM}}$ system	
s_c:	*simulect* belongs to a group of medicines called *immunosuppressants*.
s_t:	*simulect* fait parti d ' une classe de médicaments appelés *immunosuppresseurs*.
Output:	**zeffix fait parti d ' une classe de médicaments appelés antiviraux.**
$\mathrm{EBMT_{TM}} + \mathrm{SMT} +$ *index* system	
s_c:	*diacomit* belongs to a group of medicines called *antiepileptics*.
s_t:	*diacomit* appartient à un groupe de médicaments appelés *antiépileptiques*.
Output:	**zeffix appartient à un groupe de médicaments appelés antiviraux.**

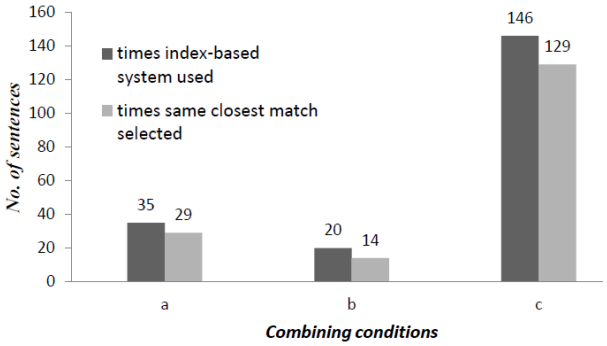


Figure 3: Number of times $\mathrm{EBMT_{TM}} + \mathrm{SMT} +$ *index* used in the hybrid system and the number of times the same closest-matching sentences are selected by the systems. a=FMS>0.85, b=FMS>0.85 & EqUS and c=FMS>0.80 OR (FMS>0.70 & EqUS)

improvement in translation time, and BLEU scores remained the same or increased. Due to the selection of different closest-matching sentence s_c, sometimes the system produces better quality translation which increases the system level BLEU score. Table 7 shows one such En—Fr example where an index-based technique produced a better translation than the baseline ($\mathrm{EBMT_{TM}} + \mathrm{SMT}$) system.

7 Conclusion

Our experiments show that EBMT approaches work better compared to the SMT-based system for certain sentences when a high fuzzy match score is obtained for the input sentence with the example-base. Thus a feature-based combination of EBMT- and SMT-based systems produces better translation quality than either of the individual systems. Integration of a SMT technology-based sub-sentential TM with the EBMT framework ($\mathrm{EBMT_{TM}}$) has improved translation quality in our experiments.

Our baseline $\mathrm{EBMT_{TM}}$ system is a runtime approach which has high time complexity when using a large example-base. We found that the integration of IR-based indexing substantially improves run time without affecting BLEU score. So far our systems have been tested using moderately sized example-bases from a closed domain corpus. In our future work, we plan to use a much larger example-base and wider-domain corpora.

Acknowledgments

This research is supported by Science Foundation Ireland (Grants 07/CE/I1142, Centre for Next Generation Localisation).

References

S. Armstrong, C. Caffrey, M. Flanagan, D. Kenny, M. O'Hagan and A. Way. 2006. Improving the Quality of Automated DVD Subtitles via Example-Based Machine Translation. Translating and the Computer **28**, [no page number], London: Aslib, UK.

E. Biçici and M. Dymetman. 2008. Dynamic Translation Memory: Using Statistical Machine Translation to Improve Translation Memory. In Gelbukh, Alexander F., editor, In *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing),* volume 4919 of *Lecture Notes in Computer Science,* pp 3-57 Springer Verlag.

J. Bourdaillet, S. Huet, F. Gotti, G. Lapalme and P. Langlais. 2009. Enhancing the bilingual concordancer TransSearch with word-level alignment. In *Proceedings, volume 5549 of Lecture Notes in Artificial Intelligence: 22nd Canadian Conference on Artificial of Intelligence (Canadian AI 2009),* Springer-Verlag, pp. 27-38.

R. D. Brown. 2011. The CMU-EBMT machine translation system. *Machine Translation*, **25**(2):179–195.

I. Cicekli and H. A. Güvenir. 2001. Learning translation templates from bilingual translation examples. *Applied Intelligence*, **15**(1):57–76.

S. Dandapat, S. Morrissey, A. Way and M.L. Forcada. 2011. Using Example-Based MT to Support Statistical MT when Translating Homogeneous Data in Resource-Poor Settings. In *Proceedings of the 15th Annual Meeting of the European Association for Machine Translation (EAMT 2011),* pp. 201-208. Leuven, Belgium.

S. Dandapat, M.L. Forcada, D. Groves, S. Penkale, J. Tinsley and A. Way. 2010. OpenMaTrEx: a free/open-source marker-driven example-based machine translation system. In *Proceedings of the 7th International Conference on Natural Language Processing (IceTAL 2010),* pp. 121-126. Reykjavík, Iceland.

D. Groves and A. Way. 2006. Hybridity in MT: Experiments on the Europarl Corpus. In *Proceedings of the 11th Conference of the European Association for Machine Translation (EAMT 2006),* pp. 115-124. Oslo, Norway.

M. Islam, J. Tiedemann and A. Eisele. 2010. English–Bengali Phrase-based Machine Translation. In *Proceedings of the 14th Annual Conference of the European Association of Machine Translation, (EAMT 2010),* [no page number], Saint-Raphaël, France.

M. Khalilov, J.A.R. Fonollosa, I. Skadina, E. Bralitis and L. Pretkalnina. 2010. English–Latvian SMT: the Challenge of Translating into a Free Word Order Language. In *Proceedings of the 2nd International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU 2010),* [no page number], Saint-Raphaël, France.

P. Koehn. 2010. *Statistical Machine Translation*, Cambridge University Press, Cambridge, UK.

P. Koehn, 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 *Proceedings of the Demonstration and Poster Sessions at the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007),* pp. 177-180. Prague, Czech Republic.

P. Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004),* pp. 388-395. Barcelona, Spain.

P. Koehn and J. Senellart. 2004. Convergence of Translation Memory and Statistical Machine Translation. In *Proceedings of the AMTA workshop on MT Research and the Translation Industry,* pp. 21-23. Denever, CO.

P. Langlais, G. Lapalme and M. Loranger. 2002. Development-evaluation cycles to boost translator's productivity. *Machine Translation*, **15**(4):77–98.

Y. Lepage and E. Denoual. 2005. Purest ever example-based machine translation: Detailed presentation and assessment. *Machine Translation*, **19**(3-4):251–282.

V. I. Levenshtein. 1965. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Doklady Akademii Nauk SSSR,* **163**(4):845-848., English translation in Soviet Physics Doklady,**10**(8), 707-710.

C. D. Manning, P. Raghavan and H. Schütze. 2008. *Introduction to Information Retrieval.* Cambridge University Press, Cambridge, UK.

M. Nagao. 1984. A Framework of a Machine Translation between Japanese and English by Analogy Principle. In Elithorn, A. and Banerji, R., editors, *Artificial Human Intelligence,* pp. 173–180, North-Holland, Amsterdam.

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

K. Papineni, S. Roukos, T. Ward and W. J. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, (ACL 2002),* pp. 311–318, Philadelphia, PA.

A. B. Phillips. 2011. Cunei: open-source machine translation with relevance-based models of each translation instance. *Machine Translation*, **25**(2):161–177.

M. Simard and P. Isabelle. 2009. Phrase-based Machine Translation in a Computer-assisted Translation Memory. In *Proceedings of the 12th Machine Translation Summit, (MT Summit XII),* pp. 120–127, Ottawa, Canada.

M. Simard. 2003. Translation spotting for translation memories. In *Proceedings of the HLT-NAACL 2003, Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond,* pp. 65–72, Edmonton, Canada.

H. Somers. 2003. An Overview of EBMT. In M. Carl and A. Way , editors, *Recent Advances in Example-based Machine Translation,* pp. 3-57, Kluwer Academic Publishers, Dordrecht, The Netherlands.

J. Tiedemann and L. Nygaard. 2009. News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces, in N. Nicolov, K. Bontcheva, G. Angelova, R. Mitkov. (eds.), *Recent Advances in Natural Language Processing*, **V**:237–248, John Benjamins, Amsterdam, The Netherlands.

E. Ukkonen. 1983. On Approximate String Matching. In *Proceedings of International Conference on Foundations of Computing Theory, (FCT 1983),* pp. 487–496, Borgholm, Sweden.

R. Wagner and M. Fischer. 1974. The String-to-String Correction Problem. *Journal of the Association for Computing Machinery*, **21**:168–173.