

Using Example-Based MT to Support Statistical MT when Translating Homogeneous Data in a Resource-Poor Setting

Sandipan Dandapat¹, Sara Morrissey¹, Andy Way¹, Mikel L. Forcada²

¹CNGL, School of Computing

Dublin City University, Glasnevin, Dublin 9, Ireland

{sdandapat, smorri, away}@computing.dcu.ie

²Departament de Llenguatges i Sistemes Informàtics

Universitat d'Alacant, E-03071 Alacant, Spain

mlf@dlsi.ua.es

Abstract

In this paper, we address the issue of applying example-based machine translation (EBMT) methods to overcome some of the difficulties encountered with statistical machine translation (SMT) techniques. We adopt two different EBMT approaches and present an approach to augment output quality by strategically combining both EBMT approaches with the SMT system to handle issues arising from the use of SMT. We use these approaches for English to Turkish translation using the IWSLT09 dataset. Improved evaluation scores (4% relative BLEU improvement) were achieved when EBMT was used to translate sentences for which SMT failed to produce an adequate translation.

1 Introduction

Over the past two decades, machine translation (MT) has shown very promising results, mostly using statistical machine translation (SMT) techniques. However, a large number of languages exist which suffer from the scarcity of parallel corpora, e.g. Indic languages, sign languages etc. SMT approaches have yielded low translation quality for these poorly resourced languages (Khalilov et al., 2010). It is often the case that domain-specific translation is required to tackle the issue of scarce resources, but it can still suffer from very low accuracy within the SMT framework, even for homogeneous domains (Dandapat et al., 2010). Although SMT and EBMT are both data-driven approaches to MT, both of them have their own advantages and lim-

itations. Typically, an SMT system works well with significant amounts of training data. In contrast, an EBMT approach can be developed with a limited example-base (Somers, 2003); also, as with any other data-driven system, an EBMT system works well when training and test sets are quite close in nature (Marcu, 2001). This is because EBMT systems reuse the segments of test sentences that can be found in the source side of the example-base at runtime.

Keeping these points in mind, it is important to develop an MT system of reasonably good quality based on limited amounts of data. In this direction, we are inspired to examine different EBMT approaches which can handle the problem of data sparseness. It is often the case that EBMT systems produce a good translation where SMT fails and vice versa. In order to harness the advantages of both approaches, we use a careful combination of both EBMT and SMT to improve translation accuracy.

We adopt two alternative approaches to tackle the above problems. First we adopt a compiled approach to EBMT which essentially produces translation templates during the training stage, based on the description in (Cicekli and Güvenir, 2001). Our second attempt presents a novel way of integrating translation memory (TM) into an EBMT system. Starting with the user's TM as a training set, additional sub-sentential translation units (TUs) are extracted based on the word alignments produced by an SMT system. These sub-sentential TUs are used both for alignment and recombination after the closest matching example to the input is found in the matching stage of our EBMT system.

The rest of the paper is organized as follows. The next section presents related research in the area. Section 3 describes some related issues behind our particular approach. Section 4 describes

the detail of our EBMT-based approaches. Section 5 presents the experimental setup, the data and the results obtained with different experiments. Section 6 presents our observations with an analysis of errors. We conclude in section 7 with some avenues for future work.

2 Related Work

The EBMT framework was proposed by Nagao (Nagao, 1984) as *translation by analogy*. An EBMT system relies on past translations to derive the target output for a given input and performs the translation in three steps: matching, alignment and recombination (Somers, 2003). The two main approaches to EBMT are distinguished by the inclusion or exclusion of a preprocessing/training stage. 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. Approaches that incorporate a training stage are commonly called “compiled approaches”, as training consists of compiling units below the sentence level. Cicekli and Güvenir (2001) proposed an approach generalized over sequences of words. The underlying assumption is that given two parallel sentence pairs, translation templates can be learned based on the similarities in both the source and target sides. The same applies to the differing parts between two parallel sentences. Generalization in this approach consists of replacing the similar or differing sequences with variables and producing a set of translation templates (including *atomic translation templates* containing no variables). These translation templates are later used to translate new input sentences. Prior to the above approach, other research was carried out to learn translation templates based on syntactic generalization, e.g. (Kaji et al., 1992). A recent work has also focused on morphological generalization to learn EBMT templates (Phillips et al., 2007).

EBMT is often linked with a related technique, namely TM. A TM essentially stores source- and target-language translation pairs (called *translation units*, TUs) for effective reuse of the previous translations. TM is often used to store examples for EBMT systems. It is also widely used in computer-aided translation (CAT) systems to assist professional translators. EBMT systems first find the example (or a set of ex-

amples) from the TM which most closely matches the source-language string to be translated (Somers, 2003). After retrieving a set of examples, with associated translations, EBMT systems automatically extract the translation of the suitable fragments and combine them to produce a grammatical target output. On the other hand, CAT systems segment the input text to be translated and compare each segment against the TUs in the TM (Bowker, 2002). CAT systems produce one or more target equivalences for the source segment and professional translators select and recombine them (perhaps with modification) to produce the desired translation. Both EBMT and CAT systems are developed based on a similar premise but in an EBMT approach, selection and recombination are done automatically to produce the translation without the help of a professional translator.

Phrase-based SMT systems (Koehn, 2010), produce a source–target aligned subsentential phrase table which can be adapted as an additional TM to a CAT environment (Simard, 2003; Bourdaillet et al., 2009). SMT phrases have also been used to populate the knowledge database of an EBMT system (Groves and Way, 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 matching 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. Our approach attempts to integrate the subsentential TM obtained using SMT techniques within an EBMT system.

3 Related Issues

3.1 Type of Corpora

Both EBMT and SMT are data-driven approaches to MT which need machine-readable corpora as a prerequisite. The size and type of corpus is also important for adopting a particular approach to MT. In order to deal with less-resourced homogeneous data, we used the IWSLT09 English–Turkish data for our experiments. While the IWSLT09 training data is quite small, we found that the corpus is comprised of very similar domain-specific sentences, as illustrated in (1) and (2).

- (1) *a. I'd like to see that camera on the shelf.*
b. I'd like to have it parted on the left.
- (2) *a. Have you ever seen a Japanese movie?*
b. Have you ever tried Japanese food?

The portions in italics are the only differences between (a) and (b). Thus, it might be helpful to reuse the translation of the common part while translating a new sentence. The above observation leads us to reuse some parts of the sentence which are common with the closest sentence in the example-base in our EBMT system.

3.2 Building a Sub-sentential TM

Building a high-quality TM is an expensive and time-consuming process. As we have no access to any TM for English–Turkish beyond the data mentioned in section 3.1, we decided to build an auxiliary sub-sentential TM automatically from our small training corpus. We used Moses¹ to automatically build this TM. Based on Moses word alignment (using GIZA++) and a phrase table, we construct the additional TM for further use. First, we add entries to the TM based on the aligned phrase pairs from the Moses phrase table. A source phrase may have multiple target equivalents. We keep all target equivalents in a sorted order based on the phrase translation probability. This helps us in the matching procedure, but during recombination we only consider the most probable target equivalent (section 4.2). Furthermore, we add entries in the TM based on the source-to-target word-aligned file. We keep the multiple target equivalents for a source word in sorted order. This essentially adds source- and target-language equivalent word pairs into the TM. Table 1 shows some of the TUs in the TM.

Source (English)	Target (Turkish)
Example entries in TM from Moses phrase table	
i don't like it	{{"sevmedim", "bunu sevmedim"}}
i can't sleep well .	{{"iyi uyuyamıyorum ."}}}
a hotel	{{"bir otel", "bir otelde", "otel"}}}
Example entries in TM from Moses word-aligned file	
coffees	{{"kahve"}}
fair	{{"fuar", "bayanımı", "ortalama"}}
helps	{{"vücutun", "yardım", "eder"}}

Table 1: Source-target translation equivalents in TM.

4 Our Approaches

We adopt two different approaches in order to estimate the relative performance of both under

¹ Moses (<http://www.statmt.org/moses/>) is an SMT tool that includes routines to automatically train translation models for any language pair. The *lexical translation table* (of training step 4) and *score phrases* (of training step 6) are used to build our TMs.

the same experimental setup. Our first approach adopts the method of automatically learning translation templates and reusing them for translating new input (Cicekli and Güvenir, 2001). Our second approach describes a novel way of using EBMT techniques with a TM to tackle the problem of data sparseness.

4.1 Generalized Translation Template-based EBMT

We have implemented our generalized translation-template-based EBMT system based on the description given in (Cicekli and Güvenir, 2001). We shall refer to this as **GEBMT** in the rest of the paper. In this approach, we have a clear separation between the learning and decoding modules. The learning phase of the GEBMT system learns templates from a sentence-aligned bitext by studying the similarities and differences between the two example pairs. Consider the following two source and target English–Turkish examples in (3) :

- (3) *a. I will drink orange juice: portakal suyu içeceğim*
b. I will drink coffee: kahve içeceğim

Clearly, these two examples share the word sequence *I will drink* and differ in the word sequence *orange juice* and *coffee*. Similarly in the target side the similar part is *içeceğim* and differing parts are *portakal suyu* and *kahve*. Based on this observation, the following subsentential alignments in (4) can be captured:

- (4) *I will drink : içeceğim;*
coffee : kahve;
orange juice : portakal suyu

By substituting the similar and differing sequence with variables, the templates in (5) can be obtained:

- (5) *a. I will drink X^s : X^t içeceğim*
b. X^s orange juice : portakal suyu X^t
c. X^s coffee : kahve X^t

These generalized templates are added to the example-base along with the atomic templates in (4). More details of the learning algorithm can be found in Cicekli and Güvenir (2001:p. 58).

4.1.1 Decoding

During decoding we deviate from the approach described in Cicekli and Güvenir (2001). Our alternative approach is shown in Figure 1.

After learning the templates we assign a probabilistic score to each translation template ($T_i: s_i \rightarrow t_i$) using the counts in (i).

$$(i) p(t_i|s_i) = \text{count}(s_i \rightarrow t_i) / \text{count}(s_i)$$

We also assign a weight factor (w) to each translation template (T_i) during the runtime decoding process, as in step 8 of the algorithm; the factor represents the ratio of surface words in the source part of the pattern to the length of the partial translation Y . These two factors (p and w) are multiplied to assign a translation score (q) to each output translation in the potential translation set (step 9 of the algorithm). In step 5, untranslated(Y) returns the set of untranslated substrings in the partial translation Y . In step 7, the function substitute($Y, Z, s \rightarrow t$) generates a new partial translation where Z is substituted by t so that those parts of Z matched by variables in s are copied to the position of their corresponding variables in t .

Algorithm 1 decoding(X, T, M, N)

In: source sentence X ,
translation template set T ,
number of top-ranking translations M ,
number of live hypotheses during decoding N
Out: set of the M best translations and their score H

- 1: $\{S$ is the set of partial translations containing pairs (partial translation, score)}
- 2: $S \leftarrow (X, 1)$ {initialise set to contain the source sentence and score=1}
- 3: **repeat**
- 4: $S' \leftarrow \emptyset$
- 5: **for all** $(Y, q) \in S$ **and** $Z \in \text{untranslated}(Y)$ **do**
- 6: **for all** $s \rightarrow t \in T$ such that s matches Z **do**
- 7: $Y' = \text{substitute}(Y, Z, s \rightarrow t)$
- 8: $w = \text{numSurfaceWords}(s) / \text{length}(Y)$
- 9: $q \leftarrow q \times p \times w$
- 10: $S' \leftarrow S' \cup \{(Y', q')\}$
- 11: **end for**
- 12: **end for**
- 13: **for all** $(Y', q') \in S'$ **do**
- 14: **if** untranslated(Y') $\neq \emptyset$ **then**
- 15: $S' \leftarrow S' - (Y', q')$ {remove complete translation for further processing}
- 16: $H \leftarrow \text{top}(H \cup \{(Y', q')\}, M)$ {add to ordered set of completed translations H }
- 17: **end if**
- 18: $S \leftarrow \text{top}(S', N)$ {only N partial translations are considered in the next iteration}
- 19: **end for**
- 20: **until** $S \neq \emptyset$

Figure 1: Decoding procedure of the GEBMT system

4.2 EBMT using TM

Like all other EBMT systems, our particular approach comprises three stages: *matching*, *alignment* and *recombination*.

4.2.1 Matching

The first step in an EBMT system is to find source-language examples that closely match the input sentence. In particular, in our approach, we

find *the closest sentence* (s_c) from the example-base for the input sentence (s), as in (ii).

$$(ii) s_c = \arg \max_{s_i} \text{score}(s, s_i)$$

We used a word-based edit distance metric (Wagner and Fischer, 1974) to find the closest-matching sentence from the example-base, s_i , based on equation (iii).

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

where $|x|$ denotes the length (in words) of a sentence, and $\text{ED}(x, y)$ refers to the word-based edit distance between x and y .

Based on the above fuzzy scoring criteria, we are able to choose the closest match for the input sentence to be translated. For the input sentences in (6), the corresponding closest fuzzy-matched sentence from the example-base is given in (7).

(6) *i 'd like a present for my mother .*

(7) *i 'd like a shampoo for greasy hair .*

Then we consider the associated translation (t_c) (8) of the closest matching source sentence (7),

(8) *yağlı saçlar için bir şampuan istiyorum .*

GREASY HAIR FOR ONE SHAMPOO 'D-LIKE

to build a skeleton for the translation of the input sentence (6) as described in the following two subsections.

4.2.2 Alignment

After matching and retrieving an example with its associated translation, the next step is to extract from that translation the non-matching fragments. In order to do that, we align the three sentences: the input (s), the closest source-side match (s_c), and its target equivalent (t_c).

First, we mark the mismatch portion between s and s_c while computing the edit distance in equation (iii). This is shown in (9) with angle brackets (the numbers within the angle brackets index the mismatched segments). Then we align the mismatch in s_c with its associated translation t_c using TM. Based on the source-target aligned pair from the TM, we mark the mismatch segment in the t_c as in (9c). The portions marked with angle brackets in (9c) are aligned with the mismatch portion in (9b). Here also, the numbers within the angle brackets in t_c indicate the mapping between the segments with s_c .

(9) a. s : *i 'd like a <0:present> for <1:my mother> .*

b. s_c : *i 'd like a <0:shampoo> for <1:greasy hair> .*

c. t_c : *<1:yağlı saçlar> için bir <0:şampuan> istiyorum .*

With the help of the above matching method, in the recombination step, we will replace the seg-

ments within the angle brackets while keeping the remaining matched fragments unchanged.

4.2.3 Recombination

The final step of our EBMT approach is recombination. We add or substitute segments from the input sentence (s) into the skeleton of the translation equivalent derived from t_c . From (9), we need to replace the two segments in (9c) $\{yağlı saçlar$ (greasy hair) and $şampuan$ (shampoo)} with two corresponding source segments in (9a) $\{my\ mother$ and $present\}$ to produce a target equivalent. Thus keeping the mapping, we produce (10):

(10) $\langle 1:my\ mother \rangle$ için bir $\langle 0:present \rangle$ istiyorum.

Note that there might not be a one-to-one correspondence between s and t_c . If there is some extra segment in s which does not have any mapping in t_c , then we add the new segment from s into the target equivalent t_c . If there is an extra segment in t_c , then we simply delete it from the target translation. Thus we produce the target equivalent in (10) after adding/deleting/substituting segments from the input sentence to be translated (s) with the skeleton translation (t_c). Then, we translate the new untranslated segments in (10). We translate these new segments using the subsentential TM to produce the translation of the untranslated segments. The detail of the algorithm² is given in Figure 2.

Algorithm 2 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
-

Figure 2: Algorithm to translate each subsentential segment X corresponding to the mismatched segments between the source sentence s and its closest match s_c .

Replacing the untranslated segment in (10) with the translation obtained using TM , we derive the output translation of the original input sentence in (11):

²Our future work includes a more efficient implementation of this algorithm using dynamic programming.

(11) $\langle annem \rangle$ için bir $\langle hediye \rangle$ istiyorum .

Note that unknown words are left untranslated, which is the case for most MT techniques. Incorrect translations may be expected due to incorrect word/phrase alignments.

5 Experiments

First, we conduct two different experiments to estimate the baseline accuracy of our approach for the English–Turkish translation task. We use the Moses **SMT** system as a baseline and compare the results with our approach.

First we conduct an experiment using our **GEBMT** system (cf. Section 4.1).

We conduct a second experiment based on the matching step (section 4.2) of the EBMT system: 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 **EBMT** in the experiments below. We will consider this as the baseline accuracy for our EBMT system which uses the TM approach.

After obtaining the skeleton translation through matching and alignment steps (section 4.1), in the *recombination step*, we use the subsentential TM to translate any unmatched segments as described in Figure 2. We call this **EBMT_{TM}**.

We found that there are cases where the EBMT systems (GEBMT or EBMT_{TM}) produce the correct translation but SMT fails and vice-versa. In order to further improve translation accuracy, we use a combination of EBMT and SMT, and so we conducted two more experiments.

We assume that the translations of an input sentence s produced by GEBMT and SMT are respectively $T_{GEBMT}(s)$ and $T_{SMT}(s)$. We also have the translation score (q) for each output produced by the GEBMT system. If the value of q is greater than some threshold, we rely on the output $T_{GEBMT}(s)$; otherwise we take the output from $T_{SMT}(s)$. We conduct experiments with the threshold for q varying from 0.3 to 0.9 to see the relative effect of the threshold. We call this system **GEBMT_{score>x}+SMT**.

In a similar way, we combine the EBMT_{TM} with SMT. Here we use some features to decide whether we will rely on the output produced by the EBMT_{TM} system. These features include the *fuzzy match score* (**FMS**) (as in (iii)) and the equality in the number of unmatched segments in s , s_c , and t_c (**EqUS**), as in example (9). We refer to this system as **EBMT_{TM}+SMT**.

5.1 Data used for the Experiments

For all experiments we used the English–Turkish corpus from IWSLT09. The training data consists of 19,972 parallel sentences. We used the 414-sentence IWSLT09 development set as our test set. We conduct experiments with varying training data sizes randomly drawn from the whole training data to understand the relative performance of the different systems.

5.2 Results

We used BLEU (Papineni et al., 2002) and NIST (Doddington, 2002) for automatic evaluation of our systems. Table 2 shows the results obtained with different GEBMT systems along with the amount of training data used to train the systems.

System	BLEU(%)	NIST
Training Data: 1242 sentences		
SMT	7.63	2.98
GEBMT	6.80	2.78
GEBMT _{score>0.3} +SMT	7.96	2.98
Training Data: 2184 sentences		
SMT	10.72	3.51
GEBMT	7.21	3.07
GEBMT _{score>0.9} +SMT	10.83	3.52
GEBMT _{score>0.8} +SMT	10.99	3.53
GEBMT _{score>0.7} +SMT	10.76	3.53
GEBMT _{score>0.6} +SMT	10.55	3.52

Table 2: System accuracies obtained by different GEBMT models. The subscript *score>x* denotes the value of translation score (q).

Note that due to the large time complexity of the training algorithm of the GEBMT approach, we conducted our GEBMT experiments with a smaller subset (1242, 2184 sentences) of the whole training data.

Table 3 shows the accuracy obtained by the different EBMT_{TM} systems described in section 4.2. Here we have the two baselines (SMT and EBMT) as described in the first two experiments in section 5.

System	BLEU (%)	NIST
Training Data: 19972 sentences		
SMT	23.59	4.85
EBMT	15.60	3.34
EBMT _{TM}	20.08	4.41

Table 3: The baseline score of the two systems and the score of the EBMT_{TM} system.

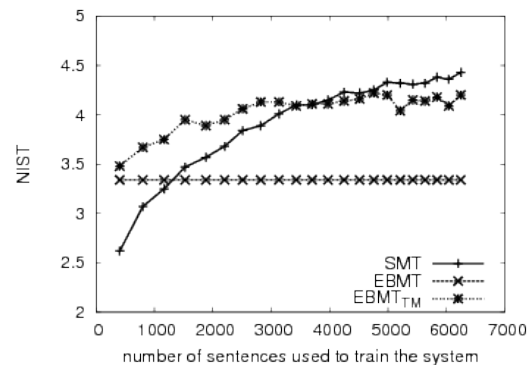
The above table shows that EBMT_{TM} has a lower system accuracy, but combining EBMT_{TM} with SMT has a different effect, as shown in Table 4. Improvements are statistically significant

(reliability of 99%) only for very high FMS (> 0.85).

System: EBMT _{TM} +SMT			
Condition	times/percentage EBMT _{TM} used	BLEU (%)	NIST
FMS>0.85	35 (8.5%)	24.22	4.89
FMS>0.8	114 (27.5%)	23.99	4.84
FMS>0.7	197 (47.6%)	22.74	4.73
FMS>0.8 OR (FMS>0.7 & EqUS)	165 (40.0%)	23.87	4.83
FMS>0.85 & EqUS	24 (5.8%)	24.41	4.9
FMS>0.8 & EqUS	76 (18.4%)	24.19	4.88
FMS>0.7 & EqUS	127 (30.7%)	24.08	4.87

Table 4: The accuracies of the combined system (EBMT_{TM}+SMT) with different combining factors. The second column indicates the number (and percentage) of sentences translated by the EBMT_{TM} system during combination.

We experiment with increasing amounts of training data to understand the performance of the EBMT_{TM} system. While selecting incremental training data, we always include the sentences which are the closest matches to the test set. This is to observe the system performance especially for a scenario where an example-base closely matching the test set is available. Figures 3(a) and (b) depict the NIST and BLEU score for different data sizes, respectively.



Figures 3(a): NIST scores obtained by 3 different systems with different data sizes.

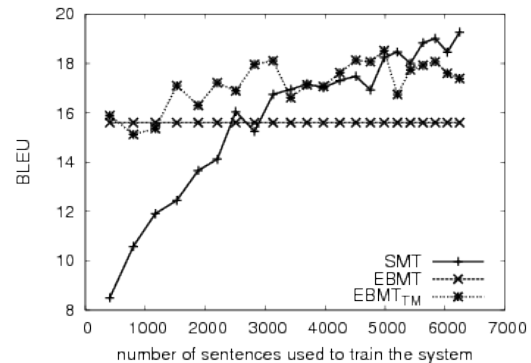


Figure 3(b): BLEU scores obtained by 3 different systems with different data sizes.

6 Observations

6.1 Summary of the Results

We find that the generalized template-based EBMT (GEBMT) has a lower accuracy on its own compared to the baseline SMT. However, while combining the GEBMT system with SMT, Table 4 shows a relative improvement of 4.3% when translation score (q) is more than 0.3 and 1242 sentences were used to train the system. Upon increasing the training data (2184 sentences), no improvements were observed for $q < 0.7$. Under the circumstances, the biggest improvement (2.5% relative BLEU points) was achieved for $q > 0.8$.

With our second experiment, we see from Figures 3(a) and 3(b) that the EBMT_{TM} system has a higher score than both baseline systems when the amount of training data is less than 5000 sentences. However, with increased data sizes, SMT performs better compared to the EBMT_{TM} system. However, there remain some sentences which are better translated by the 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 (highest improvement of 2.67% relative improvement in BLEU) compared to the baseline SMT approach. We found that if an input has a high fuzzy match score with the example-base, then the EBMT_{TM} system does better compared to SMT. We found that fuzzy match scores (FMS) over 0.8 showed an improvement over SMT with our current experimental setup. However, FMS might not be the only factor for triggering the EBMT_{TM} system (Marcu, 2001): we also consider the EqUS factor. Though an FMS over 0.7 shows no improvement in overall system accuracy, inclusion of the EqUS feature along with FMS does show improvement. Thus, the EBMT_{TM} approach is more effective if the number of unmatched segments in the source and the target is equal.

6.2 Assessment of Error Types

Errors are propagated mostly due to the wrong selection of source–target equivalences in the phrase table and lexical table which are used as the TUs in our TM. This results in some incorrect alignment in the matching step of our EBMT system. For example, in the sentence (12), the matching module gives the following alignment:

- (12) s : i have a terrible <headache> .
 s_c : i have a terrible <cough> .
 t_c : berbat bir öksürüğüm var .

In the above example, the word ‘cough’ does not have any alignment in t_c . Neither of the two target equivalents in the TM {a. öksürük (cough) b. öksürük tedavisi için (for cough treatment)} of the word ‘cough’ matches any of the words in t_c . The word aligner also fails to align any word with ‘cough’ for the sentences s_c and t_c . Furthermore, the system suffers when there is a mismatch either in the verb or in the subject of the sentence. This is because in Turkish the inflection on the verb depends on the morphological attributes of the subject.

The second type of error is propagated during the recombination step. In example (13), we have successfully matched the following fragments:

- (13) s : i want something <with shorter sleeves> .
 s_c : i want something <to cure headache> .
 t_c : <0:baş ağrısını geçiren> bir şey istiyorum.

However, in the recombination step, we need to generate the translation for the segment ‘with shorter sleeves’. We are unable to find the whole segment in the TM, and moreover none of the bigrams are present in the TM. Thus, we translate each word of the segment one by one which results in an erroneous translation ‘birlikte boydan kollu’. The most likely translation of the words ‘with’ and ‘shorter’ are ‘birlikte’ and ‘boydan’ respectively in the TM. However, this causes an error in this context as ‘boydan’ is an incorrect translation for ‘shorter’, and ‘with’ is translated to –lu in ‘kollu’.

Another common type of error occurs due to the wrong morpho-syntactic alignment and recombination. The effect can be seen in (14):

- (14) s : do you have a japanese <guidebook> ?
 s_c : do you have a japanese <magazine> ?
 t_c : japonca bir <0:derginiz> var mı ?

The word *magazine* is matched with *derginiz* (*dergi* ‘magazine’ + possessive ending) but a valid match should point out only the *dergi* part. The effect is clear when *guidebook* is translated to *rehaber kitab*: the required suffix is missing in the output. Thus, due to the rich morphology of Turkish, many morphosyntactic suffix assignment errors are generated.

7 Conclusion and Outlook

Our experiments show that EBMT approaches work better compared to the SMT-based system for certain sentences when the amount of available resource is limited. Thus a combination of

EBMT- and SMT-based systems may be expected to have a higher score than the individual systems. Integration of a sub-sentential TM with the EBMT framework has improved translation quality in our experiments.

So far our system has been tested with training and test data only of a moderate size. In order to test the scalability of our approach, we plan to use a larger dataset and a wider-domain corpus. Though the fuzzy match score has shown to be a good estimator for triggering the use of an EBMT system, we intend to find more sophisticated features to trigger the use of an EBMT system for better translation quality.

Acknowledgments: This research is supported by Science Foundation Ireland (Grants 07/CE/I1142, Centre for Next Generation Localisation and 07/W.1/I1802), ETS Walton Award for Mikel L. Forcada).

References

- 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 Intelligence (Canadian AI 2009)*, Springer-Verlag, pp. 27-38.
- L. Bowker. 2002. Computer-aided translation technology: a practical introduction. Chapter *Translation Memory Systems*, University of Ottawa Press, Ottawa, pp. 92-127.
- I. Cicekli and H. A. Güvenir. 2001. Learning translation templates from bilingual translation examples. *Applied Intelligence*. **15**(1): 57-76.
- S. Dandapat, S. Morrissey, S. K. Naskar and H. Somers. 2010. Statistically motivated example-based machine translation using translation memory. In *Proceedings of the 8th International Conference on Natural Language Processing, ICON 2010*, Kharagpur, India. pp. 168-177.
- G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram occurrence statistics. In *Proceedings of the 2nd International Conference on Human Language Technology Research (HLT 2002)*, San Diego, CA, pp 138-145.
- 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*, Oslo, Norway, pp. 115-124.
- H. Kaji, Y. Kida, and Y. Morimoto. Learning translation templates from bilingual text. In *Proceedings of the 15th [sic] International Conference on Computational Linguistics (COLING'92)*, Nantes, France, pp. 672-678.
- 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 Second International Workshop on Spoken Languages Technologies for Under-resourced Languages (SLTU 2010)*, Penang, Malaysia, pp. 87-94.
- P. Koehn. 2010. *Statistical Machine Translation*, Cambridge University Press, Cambridge, UK.
- P. Langlais, G. Lapalme and M. Loranger. 2002. TransType: 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.
- D. Marcu. 2001. Towards a unified approach to memory- and statistical-based machine translation. In *Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings*, Toulouse, France, pp. 386-393.
- M. Nagao. 1984. A framework of a mechanical translation between Japanese and English by analogy principle. In A. Elithorn and R. Benerji, editors, *Artificial and Human Intelligence*, Amsterdam: North Holland, pp. 173-180.
- K. Papineni, S. Roukos, T. Ward and W. J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Annual meeting of the Association of Computational Linguistics (ACL 2002)*, Philadelphia, PA, pp. 311-318.
- A. B. Phillips, V. Cavalli-Sforza, and R. D. Brown. 2007. Improving example based machine translation through morphological generalization and adaptation. In *Proceedings of the 9th Machine Translation Summit (MT Summit IX)*, Copenhagen, Denmark, pp. 369-375.
- 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*, Edmonton, Canada, NJ, pp. 65-72.
- H. Somers. 2003. An overview of EBMT. In M. Carl and A. Way, editors, *Recent Advances in Example-Based Machine Translation*, Kluwer Academic Publishers, Dordrecht, The Netherlands, pp. 3-57.
- R. A. Wagner and M. J. Fischer. 1974. The string to string correction problem. *Journal of the Association of Computing Machinery (JACM)*, **21**(1):168-173.