A Discriminative Framework of Integrating Translation Memory Features into SMT

Liangyou Li liangyouli@computing.dcu.ie
Andy Way away@computing.dcu.ie
Qun Liu qliu@computing.dcu.ie
CNGL Centre for Global Intelligent Content,
School of Computing, Dublin City University, Ireland

Abstract

Combining Translation Memory (TM) with Statistical Machine Translation (SMT) together has been demonstrated to be beneficial. In this paper, we present a discriminative framework which can integrate TM into SMT by incorporating TM-related feature functions. Experiments on English–Chinese and English–French tasks show that our system using TM feature functions only from the best fuzzy match performs significantly better than the baseline phrase-based system on both tasks, and our discriminative model achieves comparable results to those of an effective generative model which uses similar features. Furthermore, with the capacity of handling a large amount of features in the discriminative framework, we propose a method to efficiently use multiple fuzzy matches which brings more feature functions and further significantly improves our system.

1 Introduction

Translation Memory (TM) has been widely used to assist human translators. It provides the most similar source sentence in the database together with the target translation as the reference to a human for post-editing. As TM stores legacy translations, it can give high quality and consistent translations for repetitive materials. However, it performs badly when there are no highly similar matches in TM.

In contrast, Statistical Machine Translation (SMT) automatically learns several models, such as the translation model (from parallel data) and language model (from the target side of the parallel corpus as well as other monolingual data), and uses them to translate a new sentence. The translation is produced by maximizing a weighted combination of these models. Given a large amount of data, SMT can generate better results for unseen sentences than TM. However, unless sentence-caching is utilised, it treats a seen sentence (such as a sentence in the training data) as unseen.

Clearly, TM and SMT complement one another on matched and unmatched segments, so both are receiving increasing attention from translators and researchers, who would like to combine TM and SMT together to obtain better translation quality with methods such as system recommendation (He et al., 2010a,b) or using fragments from TM in SMT (Biçici and Dymetman, 2008; Koehn and Senellart, 2010; Ma et al., 2011; Wang et al., 2013)

This paper is focused on integrating TM into SMT to improve translation quality. We present a discriminative framework which directly integrates TM-related feature functions into SMT. In this paper, we change features extracted from TM which are defined in a generative model (Wang et al., 2013) to feature functions and add them into the phrase-based translation model. Experiments on English–Chinese and English–French tasks show that our method achieves comparable results with Wang et al. (2013), and is significantly better than the baseline phrase-based system. In addition, we present a method to incorporate multiple fuzzy matches into our system, which brings further significant improvement.

In the rest of this paper, we first introduce related work on TM and SMT combination (Section 2). Then Section 3 details our discriminative framework, TM features and the approach of using multiple fuzzy matches. Then, we provide experiments to examine our method (Section 4) and give a conclusion together with avenues for future work in Section 5.

2 Related Work

As shown in experiments (e.g. Koehn and Senellart (2010) and Wang et al. (2013)), TM can give better translation than SMT for highly matched segments; SMT is more reliable than TM for other segments. Because of such complementariness, combining TM and SMT together has been explored by some researchers in recent years.

He et al. (2010a) present a recommendation system which uses an SVM (Cortes and Vapnik, 1995) binary classifier to select a translation from the outputs of TM and SMT with the selected translation being more suitable to post-editing. They take TER (Snover et al., 2006) score as the measure of post-editing effort and use it to create training instances for SVM. He et al. (2010b) extend this work by re-ranking the N-best list of SMT and TM. However, these works are focused on sentence-level selection and thus the matched phrases in TM are not used so well.

For an input sentence, even though it does not have an exact match in the TM, there are some matched phrases which could provide useful hints for translation. Biçici and Dymetman (2008) present a dynamic TM approach which dynamically adds the longest matched non-continuous phrase and its translation in the TM to the phrase table. They show a significant improvement over both SMT and TM. However their baseline SMT system seems to perform badly (Koehn and Senellart, 2010), in which case their claims need to be considered with caution. Koehn and Senellart (2010) and Ma et al. (2011) use TM in a pipeline manner: first, identifying the matched part from the best match in the TM and merging their translation with the input; then, forcing SMT to translate the unmatched part of the input sentence. One drawback of these methods is that they do not distinguish whether a match is good or not at phrase-level.

Wang et al. (2013) propose a deep integration method by using TM information during decoding. For a phrase pair applied to an input sentence, this method extracts features from the best match in the TM, and uses pre-trained generative models to estimate one or more probabilities, and then adds them into the phrase-based system for scoring a translation. These pre-trained models are built using a factored language model (Bilmes and Kirchhoff, 2003) over sequences of features. Their experiments show significant improvement over TM, SMT and pipeline approaches. However, their work requires a rather complex process to obtain training instances for these pre-trained models, and needs to define the generative relation between different features.

3 Our Method

In this section, we present a generalized discriminative framework which can integrate TM into SMT at decoding time. Under this framework, we add features from Wang et al. (2013) into the phrase-based model as TM feature functions. In addition, we describe how to use multiple fuzzy matches efficiently to improve translation quality.

3.1 Discriminative Framework

Generally, in a state-of-the-art statistical translation framework like Moses (Koehn et al., 2007), the direct translation probability is given by a discriminative framework, as shown in Equation (1):

$$P(e \mid f) = \frac{\exp\{\sum_{m=1}^{M} \lambda_m h_m(e, f)\}}{\sum_{e'} \exp\{\sum_{m=1}^{M} \lambda_m h_m(e', f)\}} \tag{1}$$

where $h_m(e, f)$ denotes the mth feature function for target e and source f, λ_m is the weight of this feature function, and M is the number of feature functions considered.

This framework works well on pre-defined features, such as the translation model features and language model features, which are based on target e and source f. However, as is well-known, once these features have been induced, the training data (which can be a data) is disregarded in decoding. In our work, we want to maintain the possibility of consulting such TM source-target segments (with exact and fuzzy matches) at runtime.

In this paper, we argue that given a foreign sentence f, the probability of its translation e is conditioned on foreign sentence f and TM D: $P(e \mid f, D)$. When D is unavailable, it falls back to $P(e \mid f)$. Thus the discriminative model in Equation (1) could be generalized to Equation (2):

$$P(e \mid f, D) = \frac{\exp\{\sum_{m=1}^{M} \lambda_m h_m(e, f, D)\}}{\sum_{e'} \exp\{\sum_{m=1}^{M} \lambda_m h_m(e', f, D)\}} \tag{2}$$

From this, we obtain the rule in Equation (3):

$$\begin{aligned} e &= \underset{e'}{\operatorname{argmax}}\{P(e' \mid f, D)\} \\ &\simeq \underset{e'}{\operatorname{argmax}}\{P(e' \mid f, D_f)\} \\ &\simeq \underset{e'}{\operatorname{argmax}}\{\sum_{m=1}^{M} \lambda_m h_m(e', f, D_f)\} \end{aligned} \tag{3}$$

When $h_m(e', f, D_f) = \log p(e')$, this is known as the language model feature; and when $h_m(e', f, D_f) = \log p(f \mid e)$, this is known as the translation model feature. From Equation (3) we can see that, for an input sentence f, instead of using the whole TM D, we only use one or more of the matches D_f in D.

In this paper, we integrate TM into a phrase-based SMT model. In decoding, the foreign input sentence f is segmented into a sequence of I phrases $\overline{f}_1^I$, and each foreign phrase $\overline{f}_i$ is translated into a target phrase $\overline{e}_i$. Thus, a TM-related feature function can be seen as the sum of I feature functions which are based on phrase pairs, as in Equation (4):

$$\begin{aligned} h(e, f, D_f) &= h(\overline{e}_1^I, \overline{f}_1^I, D_{\overline{f}_1^I}) \\ &\simeq \sum_{i=1}^{I} h(\overline{e}_i, \overline{f}_i, D_{\overline{f}_1^I}) \end{aligned} \tag{4}$$

where $h(\overline{e}_i, \overline{f}_i, D_{\overline{f}_1^I})$ gives a value measured on the phrase pair $(\overline{e}_i, \overline{f}_i)$ and TM matches $D_{\overline{f}_1^I}$.

3.2 Fuzzy Matching

In this paper, TM-related features are extracted from the matches in the TM. For retrieving matches, we use a word-based string edit distance (Koehn and Senellart, 2010) to measure the

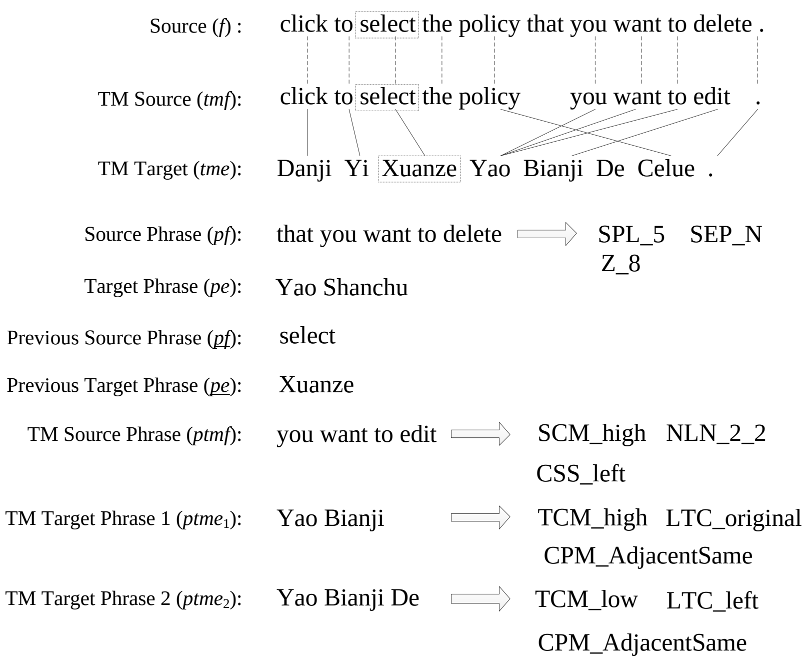

Figure 1: An example of extracting TM features. Target Chinese words are replaced by their corresponding Latin characters. The italic words in parentheses are the notions used in Section 3.3.

similarity between the input sentence and a TM instance, as in Equation (5):

$$FMS = 1 - \frac{\text{edi\_distance}(input, tm\_source)}{\max(\mid input \mid, \mid tm\_source \mid)} \tag{5}$$

During the calculation of the fuzzy match score, we also obtain a sequence of operations, including insertion, match, substitution and deletion, to convert the input sentence into a TM instance. Such operations are useful for finding the TM correspondence of a source phrase.

3.3 Translation Memory Features

In this paper, we change features from Wang et al. (2013) to TM feature functions, and add them into our phrase-based system. The value of each feature function on a sentence pair is the sum of values from features extracted on phrase pairs, as in Equation (4).

Given an input sentence f and its best match (tmf, tme) in the TM, for each phrase pair (pf, pe) applied to f, we first find its corresponding TM source phrase $ptmf$ in tmf based on operations for calculating edit-distance. Then with the help of word alignment between tmf and tme, we identify one or more TM target phrases $ptme_1^J$ in tme by extending them with unaligned words. Then we extract the following features for the phrase pair (pf, pe). Figure 1 shows an example:

- Feature set **Z_i** indicates which match in the TM is used for source phrase pf. We split fuzzy match score into 11 bins: [0, 0.1), [0.1 ,0.2), [0.2, 0.3), [0.3, 0.4), [0.4, 0.5), [0.5, 0.6), [0.6, 0.7), [0.7, 0.8), [0.8, 0.9), [0.9, 1.0), [1.0], which correspond to 11 features: $Z\_0 \cdots Z\_10$. For example, in Figure 1, $FMS(f, tmf) = 0.818$, so it goes into bin [0.8,0.9), and we add a value 1 to the feature Z_8.

- Feature set **SCM_s** represents the matching status between pf and $ptmf$. If $ptmf$ is unavailable, we add the value 1 to the feature SCM_non; if $FMS(pf, ptmf) < 0.5$, we add the value 1 to the feature SCM_low; if $FMS(pf, ptmf) > 0.5$, we add the value 1 to the feature SCM_high; and if $FMS(pf, ptmf) = 0.5$, we add the value 1 to the feature SCM_medium.

- Feature set **SPL_i** measures the length of pf. For example, if $length(pf) = 4$, we add the value 1 to the feature SPL_4. In this paper, we set maximum phrase length 7 in our system, so there are 7 features in this set.

- Feature set **SEP_b** is the indicator of whether pf is the punctuation at the end of sentence f or not. If yes, we add the value 1 to the feature SEP_Y ; otherwise, we add the value 1 to the feature SEP_N.

- Feature set **TCM_s** is the matching status between pe and $ptme_1^J$. If $ptme_1^J$ is unavailable, we add the value 1 to the feature TCM_non; otherwise, for each $ptme_i \in ptme_1^J$: if $FMS(pe, ptme_i) < 0.5$, we add the value 1 to the feature TCM_low; if $FMS(pe, ptme_i) > 0.5$, we add the value 1 to the feature TCM_high; and if $FMS(pe, ptme_i) = 0.5$, we add the value 1 to the feature TCM_medium.

- Feature set **NLN_x_y** models the matching status of context between pf and $ptmf$, where x denotes the number of matched source neighbours (left and right words) and y denotes how many of those neighbours are aligned to target words. If $ptmf$ is unavailable, we just add the value 1 to the feature NLN_non. Taking Figure 1 as an example, the left words of source phrase “that you want to delete” and TM source phrase “you want to edit” are the same and their right words are also the same, so $x = 2$. As both left and right words are aligned to target words, $y = 2$, so we add the value 1 to the feature NLN_2_2. In total, there are 6 different $< x, y >$ tuples.

- Feature set **CSS_s** describes the status of $ptme_1^J$. If $ptme_1^J$ is unavailable, we add the value 1 to the feature CSS_non; if $J = 1$, we add the value 1 to the feature CSS_single; if $J > 1$ and all phrases in $ptme_1^J$ are generated by extending only the left side, we add the value 1 to the feature CSS_left; if $J > 1$ and all phrases in $ptme_1^J$ are generated by extending only the right side, we add the value 1 to the feature CSS_right; if $J > 1$ and phrases in $ptme_1^J$ are generated by extending both sides, we add the value 1 to the feature CSS_both;

- Feature set **LTC_s** is the indicator of whether a phrase $ptme_i$ in $ptme_1^J$ is the longest or not. If $ptme_1^J$ is unavailable, we add the value 1 to the feature LTC_non; if $ptme_i$ is the phrase without being extended by unaligned words, we add the value 1 to the feature $LTC\_original$; if $ptme_i$ is only extended on its left side and has the longest left side, we add the value 1 to the feature LTC_left; if $ptme_i$ is only extended on its right side and has the longest right side, we add the value 1 to the feature LTC_right; if $ptme_i$ is extended on both sides and is the longest on both sides, we add the value 1 to the feature LTC_both; if $ptme_i$ is the one extended but not the longest one, we add the value 1 to the feature LTC_medium;

- Feature set **CPM_s** models the reordering information. if $ptmf$ is unavailable, we add the value 1 to the feature CPM_non. Otherwise, let $(\underline{pf}, \underline{pe})$ denote the last phrase pair applied to sentence f and assume the translation is generated from left-to-right. Furthermore, let $(\underline{ptmf}, \underline{ptme}_1^I)$ denote the matched TM phrase pair for $(\underline{pf}, \underline{pe})$. When both $\underline{ptme}_i$ and $ptme_j$ are available:

- if $ptme_j$ is on the right of and adjacent to $\underline{ptme}_i$,
  - if the left boundary words of pe and $ptme_j$ are the same and the right boundary words of $\underline{pe}$ and $\underline{ptme}_i$ are also the same, we add the value 1 to the feature $CPM\_AdjacentSame$.
  - otherwise, we add the value 1 to the feature $CPM\_AdjacentSubstitute$.
- if $ptme_j$ is on the right of but not adjacent to $\underline{ptme}_i$, we add the value 1 to the feature $CPM\_LinkedInterlived$.
- if $ptme_j$ is not on the right of $\underline{ptme}_i$,
  - if $ptme_j$ and $\underline{ptme}_i$ overlap, we add the value 1 to the feature $CPM\_LinkedCross$.
  - otherwise, we add the value 1 to the feature $CPM\_LinkedReversed$.

When $\underline{ptme}_i$ is unavailable and $ptme_j$ is available, we need to find the last available TM phrase pair used in the input, let it be $(\overline{ptmf}, \overline{ptme}_1^N)$, for phrase $\overline{ptme}_n$ in $\overline{ptme}_1^N$:

- if $ptme_j$ is on the right of $\overline{ptme}_n$, we add the value 1 to the feature $CPM\_SkipForward$.
- if $ptme_j$ is not on the right of $\overline{ptme}_n$,
  - if $ptme_j$ and $\overline{ptme}_n$ overlap, we add the value 1 to the feature $CPM\_SkipCross$.
  - otherwise, we add the value 1 to the feature $CPM\_SkipReversed$.

In Figure 1, the previous phrase pair is <"select","Xuanze">, and its corresponding phrase pair in the TM is indicated by a rectangle. Taking TM target phrase 1 as an example, it is to the right of and adjacent to the previous TM target phrase "Xuanze" and has the same left boundary word with the target phrase "Yao Shanchu". Furthermore, the right boundary words of the previous target phrase "Xuanze" and previous TM target phrase "Xuanze" are the same, so we use the feature $CPM\_AdjacentSame$.

3.4 Multiple Fuzzy Matches

In Section 3.3, only the best fuzzy match is used to extract features. Although we were able to find a correspondence in the TM for each source phrase, sometimes this correspondence is actually not the same as the source phrase, as shown in Figure 1. Thus we propose a method to use multiple fuzzy matches to cover as many source phrases as possible.

In this paper, besides the best match, for each source phrase we also find a TM instance which contains this phrase and has the highest fuzzy match score with the input sentence. We call such a TM instance **span-match**. Figure 2 shows an example of finding multiple matches.

Different to the best match which is estimated over the whole sentence and thus does not bias to any particular source phrase, span-match provides us with information about how a specific source phrase is used and thus may be helpful in selecting the proper target candidate. In addition, note that for a source sentence, the number of span-matches used is not fixed and has no limitation, so our method does not need to be optimized on such parameters.

When multiple fuzzy matches are considered, for each phrase pair applied to the input sentence during decoding, we extract features for it not only from the best match but also from the span-match of the source phrase. Features from span-match are the same as those defined in Section 3.3, except SPL_i and SEP_s are excluded as they are the same as features from the best match. In addition, CPM_s are not used on span-match as the current source phrase may be not using the same span-match as the last phrase. We distinguish features from the best match and the span-match by adding additional information, such as feature

Source: click to select the policy that you want to delete .
1 2 3

TM Source 1: click to select the policy you want to edit .

TM Source 2: click to select the existing policy that you want have replaced .

TM Source 3: in the policies pane , click the specific policy that you want to delete .

Figure 2: An example of finding multiple matches.

EN-ZH	sentences	words(EN)	words (ZH)
train	86,602	1,148,126	1,171,313
dev	762	10,599	10,791
test	943	16,366	16,375

EN-FR	sentences	words(EN)	words (FR)
train	765,922	20,604,865	22,401,839
dev	1,902	67,403	73,743
test	1,919	71,228	78,177

Table 1: Summary of English–Chinese (EN-ZH) and English–French (EN-FR) corpus

BFM_SCM_high, which is from the best match, and $SPAN\_SCM\_high$, which is from the span-match. In addition, we also define two more features:

- Feature **NO_SPAN_MATCH** means we cannot find a span-match for current source phrase.
- Feature **IS_SPAN_BEST** means this span match is equal (the same fuzzy match score) to the best match.

4 Experiment

4.1 Data

Our English-Chinese data set is a translation memory from Symantec, as shown in Table 1. Our English–French data is from the publicly available JRC-Acquis corpus.[1] Sentences are tokenized with scripts in Moses. We randomly select 3000 sentence pairs as dev data and 3000 as test data. We filter sentence pairs longer than 80 words in the training data and 100 words in the dev and test data. We also keep the length ratio less than or equal to 3 in all data sets. Table 1 also shows a summary of English–French corpus.

4.2 Baseline

On both language-pairs, we take the phrase-based model in Moses with default settings as our baseline. Word alignment is performed by GIZA++ (Och and Ney, 2003), with heuristic function *grow-diag-final-and* (Koehn et al., 2003). We use SRILM (Stolcke, 2002) to train a 5-gram language model on the target side of the training data with modified Kneser-Ney

[1] `http://ipsc.jrc.ec.europa.eu/index.php?id=198`

Feature Set	Feature name
Z_i	Z_0, Z_1, Z_2, Z_3, Z_4, Z_5, Z_6, Z_7, Z_8, Z_9, Z_10
SCM_s	SCM_non, SCM_high, SCM_low, SCM_medium
SPL_i	SPL_1, SPL_2, SPL_3, SPL_4, SPL_5, SPL_6, SPL_7
SEP_	SEP_Y, SEP_N
TCM_s	TCM_non, TCM_high, TCM_low, TCM_medium
NLN_x_y	NLN_2_2, NLN_2_1, NLN_2_0, NLN_1_1, NLN_1_0, NLN_0_0
CSS_s	CSS_non, CSS_single, CSS_left, CSS_right, CSS_both
LTC_s	LTC_non, LTC_original, LTC_left, LTC_right, LTC_both, LTC_medium
CPM_s	CPM_AdjacentSame, CPM_AdjacentSubstitute, CPM_LinkedInterlived, CPM_LinkedCorss, CPM_LinkedReversed, CPM_SkipForward, CPM_SkipReversed

Table 2: The list of TM features extracted on the best match in our system.

discounting (Chen and Goodman, 1996). Minimum Error Rate Training (MERT) (Och, 2003) is used to tune weights.[2] However, when TM features are incorporated, the number of features grows to more than 50 (Table 2 show the features used in our system when only best match is considered). As MERT is known to be weak when the number of features grows (Durrani et al., 2013), we use MIRA (Cherry and Foster, 2012) instead to tune weights in this case. We set the maximum iteration of MIRA to be 25. Case-insensitive BLEU (Papineni et al., 2002) is used to evaluate the translation results. Bootstrap resampling (Koehn, 2004) is also performed to compute statistical significance with 1000 iterations.

We implement Wang et al. (2013)'s method in Moses for comparison. This method needs first to train three models[3] with the factored language model toolkit (Kirchhoff et al., 2007) over the feature sequence of phrase pairs. To obtain such phrase pairs for training, we do cross-folder translation on two language pairs. For the English–Chinese task, we split the training data into 50 parts and build 50 systems with the above settings by taking each part as test data and the rest as training data. Systems are tuned via the devset for the task. For the English–French task, we do 10-cross folder training. After training the systems, forced decoding (Schwartz, 2008) is used to generate the corresponding phrase segmentation on the test data. Then features are extracted on those phrase correspondences.[4]

We also implement our method in Moses. In this paper, training data is taken as the TM data, so phrase rules from the TM are already included during translation. After the SMT models are trained, word alignment of the TM is also produced as a by-product.

4.3 Experiment Results

Table 3 shows our experiment results on two language pairs. We found that our system with TM features achieves comparable results (+0.24/+0.31 on the dev set and +0.17/-0.01 on the test set) with Wang et al. (2013) and both systems are significantly better than the baseline. After

[2]On our baseline system, MERT performs slightly better than MIRA.

[3]Three probabilities in model III which brings best performance in their paper:

$$p(TCM \mid SCM, NLN, LTC, SPL, SEP, Z)$$
$$p(LTC \mid CSS, SCM, NLN, SEP, Z)$$
$$p(CPM \mid TCM, SCM, NLN, Z)$$

[4]In the experiment, we only use two systems for feature extraction for the English–French task as the training data is significantly large.

systems	EN–ZH		EN–FR	
	dev	test	dev	test
Phrase-based SMT	52.88	44.63	61.65	61.75
+Wang's model	**54.47**	**45.72**	**62.45**	**62.44**
+TM feature	**54.71**	**45.89**	**62.76**	**62.43**
+multiple fuzzy matches	**55.48***	**46.75***	**63.38***	**63.10***

Table 3: BLEU [%] on English–Chinese (EN-ZH) and English–French (EN-FR) data. Bold figures mean that the result is significantly better than the baseline phrase-based model at $p \leq 0.01$ level. * indicates that multiple fuzzy matches significantly improves the system with TM features at $p \leq 0.01$ level.

Ranges	Sentence	Words(EN)	Words/Sentence
[0.8, 1.0)	198	3,239	16.4
[0.6, 0.8)	195	2,876	14.7
[0.4, 0.6)	318	5,358	16.8
(0.0, 0.4)	223	4,784	21.5

(a) English–Chinese

Ranges	Sentence	Words(EN)	Words/Sentence
[0.9, 1.0)	313	10,166	32.5
[0.8, 0.9)	258	7,297	28.3
[0.7, 0.8)	216	6,128	28.4
(0.6, 0.7)	156	5,195	33.3
[0.5, 0.6)	171	5,832	34.1
[0.4, 0.5)	168	5,754	34.3
[0.3, 0.4)	277	11,157	40.3
(0.0, 0.3)	360	19,699	54.7

(b) English–French

Table 4: Composition of test subsets based on fuzzy match scores on English–Chinese and English–French data.

multiple fuzzy matches are incorporated, our system shows further significant improvement (+0.76/+0.62 on dev and +0.86/+0.67 on test).

In addition, we are also interested in the performance of the systems on different fuzzy match ranges. Table 4 shows statistics on subsets of test data based on fuzzy match ranges on English–Chinese and English–French data. We see that sentences with a lower fuzzy match score (0.0-0.4) are longer.

The BLEU scores [%] for different fuzzy match ranges are shown in Figure 3. It is easy to see that our system with multiple fuzzy matches achieves best performance over most ranges. Especially on the English–Chinese task, when both Wang's model and the TM features are ineffective on the range (0.0,0.4) and [0.4,0.6), multiple fuzzy matches improve the system to give the best translation on both language pairs. However, in the highest range, Wang et al. (2013)'s method gives the best results. It seems that our system does not bias to high-scoring fuzzy match range and treat all ranges fairly.

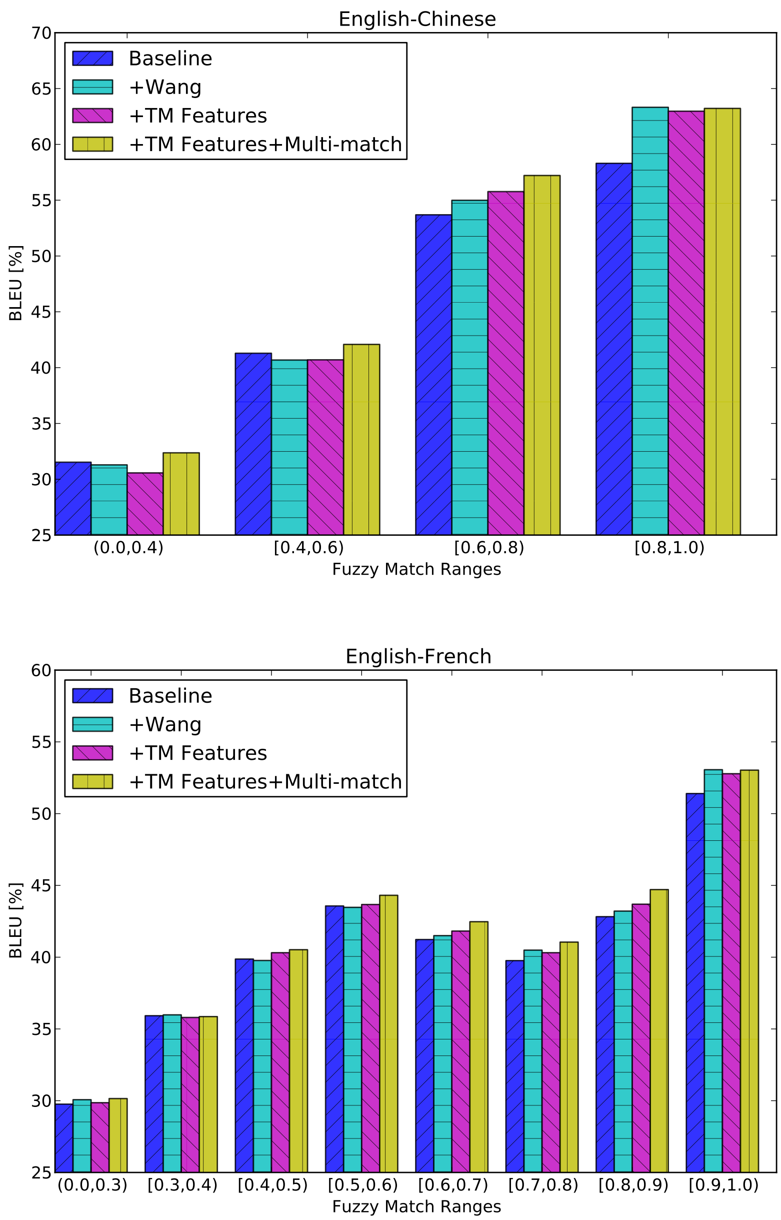

Figure 3: BLEU [%] for different fuzzy match ranges on two language pairs. The baseline is the phrase-based SMT system. The other three systems integrate different TM information into the baseline.

5 Conclusion

In this paper, we present a discriminative framework which can integrate TM into SMT. Under this framework, we add TM feature functions, which model the relation between the source sentence and TM instances, into a phrase-based SMT. In experiments on English–Chinese and English–French tasks, our method performs significantly better than the baseline phrase-based system. Furthermore, we present a method to efficiently use multiple fuzzy matches. Experiments show that this addition significantly improves our system.

Although in this paper most features are from Wang et al. (2013), our method is much simpler yet shows comparable results to their work. In addition, our method can be more easily extended with further features and integrated into other translation models, such as hierarchical phrase-based and syntax-based models. These are avenues for future work. Furthermore, as our method is SMT-centric, in the future we would also like to extend it to get the best of both worlds (SMT and TM) and .

Acknowledgements

This research has received funding from the People Programme (Marie Curie Actions) of the European Unions Seventh Framework Programme FP7/2007-2013/ under REA grant agreement no. 317471. This research is also supported by the Science Foundation Ireland (Grant 12/CE/I2267) as part of the Centre for Next Generation Localisation at Dublin City University. The authors of this paper also thank Kun Wang and Xiaofeng Wu for their help on our experiments and thank the reviewers for helping to improve this paper.

References

Biçici, E. and Dymetman, M. (2008). Dynamic Translation Memory: Using Statistical Machine Translation to Improve Translation Memory Fuzzy Matches. In *Proceedings of the 9th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 454–465, Haifa, Israel.

Bilmes, J. A. and Kirchhoff, K. (2003). Factored Language Models and Generalized Parallel Backoff. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology–short Papers*, pages 4–6, Edmonton, Canada.

Chen, S. F. and Goodman, J. (1996). An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, pages 310–318, Santa Cruz, California.

Cherry, C. and Foster, G. (2012). Batch Tuning Strategies for Statistical Machine Translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436, Montreal, Canada.

Cortes, C. and Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, 20(3):273–297.

Durrani, N., Haddow, B., Heafield, K., and Koehn, P. (2013). Edinburgh's Machine Translation Systems for European Language Pairs. In *Proceedings of the 8th Workshop on Statistical Machine Translation*, pages 114–121, Sofia, Bulgaria.

He, Y., Ma, Y., van Genabith, J., and Way, A. (2010a). Bridging SMT and TM with Translation Recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 622–630, Uppsala, Sweden.

He, Y., Ma, Y., Way, A., and Van Genabith, J. (2010b). Integrating N-best SMT Outputs into a TM System. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 374–382, Beijing, China.

Kirchhoff, K., Bilmes, J., and Duh, K. (2007). Factored Language Models Tutorial. In *UWEE Technical Report*, Department of Electrical Engineering, University of Washington.

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

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

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical Phrase-based Translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 48–54, Edmonton, Canada.

Koehn, P. and Senellart, J. (2010). Convergence of Translation Memory and Statistical Machine Translation. In *Proceedings of AMTA Workshop on MT Research and the Translation Industry*, pages 21–31, Denver, Colorado, USA.

Ma, Y., He, Y., Way, A., and van Genabith, J. (2011). Consistent Translation using Discriminative Learning - A Translation Memory-Inspired Approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1239–1248, Portland, Oregon, USA.

Och, F. J. (2003). Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41th Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 160–167, Sapporo, Japan.

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

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

Schwartz, L. (2008). Multi-Source Translation Methods. In *Proceedings of the 8th Conference of the Association for Machine Translation in the Americas*, Waikiki, Hawaii.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA.

Stolcke, A. (2002). SRILM-an Extensible Language Modeling Toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, pages 257–286, Denver, Colorado, USA.

Wang, K., Zong, C., and Su, K.-Y. (2013). Integrating Translation Memory into Phrase-Based Machine Translation during Decoding. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11–21, Sofia, Bulgaria.