Integrating Translation Memory into Phrase-Based Machine Translation during Decoding

Kun Wang[†] **Chengqing Zong**[†] **Keh-Yih Su**[‡]
[†]National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
[‡]Behavior Design Corporation, Taiwan
[†]{kunwang, cqzong}@nlpr.ia.ac.cn, [‡]kysu@bdc.com.tw

Abstract

Since statistical machine translation (SMT) and translation memory (TM) complement each other in matched and unmatched regions, integrated models are proposed in this paper to incorporate TM information into phrase-based SMT. Unlike previous multi-stage pipeline approaches, which directly merge TM result into the final output, the proposed models refer to the corresponding TM information associated with each phrase at SMT decoding. On a Chinese–English TM database, our experiments show that the proposed integrated Model-III is significantly better than either the SMT or the TM systems when the fuzzy match score is above 0.4. Furthermore, integrated Model-III achieves overall 3.48 BLEU points improvement and 2.62 TER points reduction in comparison with the pure SMT system. Besides, the proposed models also outperform previous approaches significantly.

1 Introduction

Statistical machine translation (SMT), especially the phrase-based model (Koehn et al., 2003), has developed very fast in the last decade. For certain language pairs and special applications, SMT output has reached an acceptable level, especially in the domains where abundant parallel corpora are available (He et al., 2010). However, SMT is rarely applied to professional translation because its output quality is still far from satisfactory. Especially, there is no guarantee that a SMT system can produce translations in a consistent manner (Ma et al., 2011).

In contrast, translation memory (TM), which uses the most similar translation sentence (usually above a certain fuzzy match threshold) in the database as the reference for post-editing, has been widely adopted in professional translation field for many years (Lagoudaki, 2006). TM is very useful for repetitive material such as updated product manuals, and can give high quality and consistent translations when the similarity of fuzzy match is high. Therefore, professional translators trust TM much more than SMT. However, high-similarity fuzzy matches are available unless the material is very repetitive.

In general, for those matched segments[1], TM provides more reliable results than SMT does. One reason is that the results of TM have been revised by human according to the global context, but SMT only utilizes local context. However, for those unmatched segments, SMT is more reliable. Since TM and SMT complement each other in those matched and unmatched segments, the output quality is expected to be raised significantly if they can be combined to supplement each other.

In recent years, some previous works have incorporated TM matched segments into SMT in a pipelined manner (Koehn and Senellart, 2010; Zhechev and van Genabith, 2010; He et al., 2011; Ma et al., 2011). All these pipeline approaches translate the sentence in two stages. They first determine whether the extracted TM sentence pair should be adopted or not. Most of them use fuzzy match score as the threshold, but He et al. (2011) and Ma et al. (2011) use a classifier to make the judgment. Afterwards, they merge the relevant translations of matched segments into the source sentence, and then force the SMT system to only translate those unmatched segments at decoding.

There are three obvious drawbacks for the above pipeline approaches. Firstly, all of them determine whether those matched segments

[1] We mean "sub-sentential segments" in this work.

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 11–21,
Sofia, Bulgaria, August 4-9 2013.

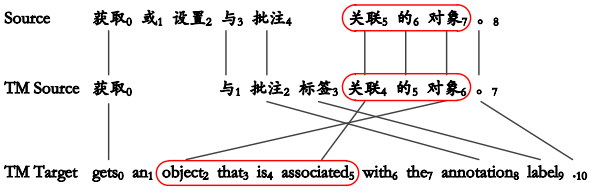

Figure 1: Phrase Mapping Example

should be adopted or not at sentence level. That is, they are either all adopted or all abandoned regardless of their individual quality. Secondly, as several TM target phrases might be available for one given TM source phrase due to insertions, the incorrect selection made in the merging stage cannot be remedied in the following translation stage. For example, there are six possible corresponding TM target phrases for the given TM source phrase "关联$_4$ 的$_5$ 对象$_6$" (as shown in Figure 1) such as "*object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", and "*an*$_1$ *object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$ *with*$_6$", etc. And it is hard to tell which one should be adopted in the merging stage. Thirdly, the pipeline approach does not utilize the SMT probabilistic information in deciding whether a matched TM phrase should be adopted or not, and which target phrase should be selected when we have multiple candidates. Therefore, the possible improvements resulted from those pipeline approaches are quite limited.

On the other hand, instead of directly merging TM matched phrases into the source sentence, some approaches (Biçici and Dymetman, 2008; Simard and Isabelle, 2009) simply add the longest matched pairs into SMT phrase table, and then associate them with a fixed large probability value to favor the corresponding TM target phrase at SMT decoding. However, since only one aligned target phrase will be added for each matched source phrase, they share most drawbacks with the pipeline approaches mentioned above and merely achieve similar performance.

To avoid the drawbacks of the pipeline approach (mainly due to making a hard decision *before* decoding), we propose several integrated models to completely make use of TM information *during* decoding. For each TM source phrase, we keep all its possible corresponding target phrases (instead of keeping only one of them). The integrated models then consider all corresponding TM target phrases and SMT preference during decoding. Therefore, the proposed integrated models combine SMT and TM at a deep level (versus the surface level at which TM result is directly plugged in under previous pipeline approaches).

On a Chinese–English computer technical documents TM database, our experiments have shown that the proposed Model-III improves the translation quality significantly over either the pure phrase-based SMT or the TM systems when the fuzzy match score is above 0.4. Compared with the pure SMT system, the proposed integrated Model-III achieves 3.48 BLEU points improvement and 2.62 TER points reduction overall. Furthermore, the proposed models significantly outperform previous pipeline approaches.

2 Problem Formulation

Compared with the standard phrase-based machine translation model, the translation problem is reformulated as follows (only based on the best TM, however, it is similar for multiple TM sentences):

$$\hat{t} = \arg\max_{t} P(t|s, [tm\_s, tm\_t, tm\_f, s\_a, tm\_a]) \quad (1)$$

Where s is the given source sentence to be translated, t is the corresponding target sentence and $\hat{t}$ is the final translation; $[tm\_s, tm\_t, tm\_f, s\_a, tm\_a]$ are the associated information of the best TM sentence-pair; tm_s and tm_t denote the corresponding TM sentence pair; tm_f denotes its associated fuzzy match score (from 0.0 to 1.0); s_a is the editing operations between tm_s and s; and tm_a denotes the word alignment between tm_s and tm_t.

Let $\bar{s}_{a(k)}$ and $\bar{t}_k$ denote the *k*-th associated source phrase and target phrase, respectively. Also, $\bar{s}_{a(1)}^{a(K)}$ and $\bar{t}_1^K$ denote the associated source phrase sequence and the target phrase sequence, respectively (total K phrases without insertion). Then the above formula (1) can be decomposed as below:

$$\begin{aligned}
\hat{t} &= \arg\max_{t} P(t|s, tm_s, tm_t, tm_f, s_a, tm_a) \\
&= \arg\max_{t} \sum_{[\bar{s}_1^K = s, \bar{t}_1^K = t]} P(\bar{t}_1^K, \bar{s}_{a(1)}^{a(K)} | s, tm_s, tm_t, tm_f, s_a, tm_a) \\
&\triangleq \arg\max_{t} \max_{[\bar{s}_1^K = s, \bar{t}_1^K = t]} \left\{ \begin{matrix} P(\bar{t}_1^K | \bar{s}_{a(1)}^{a(K)}, tm_s, tm_t, tm_f, s_a, tm_a) \\ \times P(\bar{s}_1^K | s) \end{matrix} \right\}
\end{aligned} \quad (2)$$

Afterwards, for any given source phrase $\bar{s}_{a(k)}$, we can find its corresponding TM source phrase $tm\_\bar{s}_{a(k)}$ and all possible TM target phrases (each of them is denoted by $tm\_\bar{t}_{a(k)}$) with the help of corresponding editing operations s_a and word alignment tm_a. As mentioned above, we can have six different possible TM target phrases for the TM source phrase "关联$_4$ 的$_5$ 对象$_6$". This

is because there are insertions around the directly aligned TM target phrase.

In the above Equation (2), we first segment the given source sentence into various phrases, and then translate the sentence based on those source phrases. Also, $\overline{s}_{a(1)}^{a(K)}$ is replaced by $\overline{s}_1^K$, as they are actually the same segmentation sequence. Assume that the segmentation probability $P(\overline{s}_1^K|s)$ is a uniform distribution, with the corresponding TM source and target phrases obtained above, this problem can be further simplified as follows:

$$
\begin{aligned}
&P(\overline{t}_1^K|\overline{s}_{a(1)}^{a(K)}, tm\_s, tm\_t, tm\_f, s\_a, tm\_a) \\
&= \sum_{tm\_\overline{t}_{a(1)}^{a(K)}} P(\overline{t}_1^K, tm\_\overline{t}_{a(1)}^{a(K)} \Big| \overline{s}_{a(1)}^{a(K)}, tm\_\overline{s}_{a(1)}^{a(K)}, tm\_t, z) \\
&\approx \max_{tm\_\overline{t}_{a(1)}^{a(K)}} P(\overline{t}_1^K, tm\_\overline{t}_{a(1)}^{a(K)} \Big| \overline{s}_{a(1)}^{a(K)}, tm\_\overline{s}_{a(1)}^{a(K)}, tm\_t, z) \\
&\approx \max_{tm\_\overline{t}_{a(1)}^{a(K)}} P(\overline{t}_1^K, M_1^K \Big| \overline{s}_{a(1)}^{a(K)}, L_1^K, z) \\
&\approx P(\overline{t}_1^K|\overline{s}_{a(1)}^{a(K)}) \times \prod_{k=1}^{K} \max_{tm\_\overline{t}_{a(k)}} P(M_k|L_k, z)
\end{aligned}
\tag{3}
$$

Where M_k is the corresponding TM phrase matching status for $\overline{t}_k$, which is a vector consisting of various indicators (e.g., Target Phrase Content Matching Status, etc., to be defined later), and reflects the quality of the given candidate; L_k is the linking status vector of $\overline{s}_{a(k)}$ (the aligned source phrase of $\overline{t}_k$ within $\overline{s}_1^K$), and indicates the matching and linking status in the source side (which is closely related to the status in the target side); also, z indicates the corresponding TM fuzzy match interval specified later.

In the second line of Equation (3), we convert the fuzzy match score tm_f into its corresponding interval z, and incorporate all possible combinations of TM target phrases. Afterwards, we select the best one in the third line. Last, in the fourth line, we introduce the source matching status and the target linking status (detailed features would be defined later). Since we might have several possible TM target phrases $tm\_\overline{t}_{a(k)}$, the one with the maximum score will be adopted during decoding.

The first factor $P(\overline{t}_1^K|\overline{s}_{a(1)}^{a(K)})$ in the above formula (3) is just the typical phrase-based SMT model, and the second factor $P(M_k|L_k, z)$ (to be specified in the Section 3) is the information derived from the TM sentence pair. Therefore, we can still keep the original phrase-based SMT model and only pay attention to how to extract useful information from the best TM sentence pair to guide SMT decoding.

3 Proposed Models

Three integrated models are proposed to incorporate different features as follows:

3.1 Model-I

In this simplest model, we only consider *Target Phrase Content Matching Status* (TCM) for M_k. For L_k, we consider four different features at the same time: *Source Phrase Content Matching Status* (SCM), *Number of Linking Neighbors* (NLN), *Source Phrase Length* (SPL), and *Sentence End Punctuation Indicator* (SEP). Those features will be defined below. $P(M_k|L_k, z)$ is then specified as:

$$P(M_k|L_k, z) \triangleq P(TCM_k|SCM_k, NLN_k, SPL_k, SEP_k, z)$$

All features incorporated in this model are specified as follows:

TM Fuzzy Match Interval (z): The *fuzzy match score* (FMS) between source sentence s and TM source sentence tm_s indicates the reliability of the given TM sentence, and is defined as (Sikes, 2007):

$$FMS(s, tm\_s) = 1 - \frac{\text{Levenshtein}(s, tm\_s)}{\max(|s|, |tm\_s|)}$$

Where $\text{Levenshtein}(s, tm\_s)$ is the word-based Levenshtein Distance (Levenshtein, 1966) between s and tm_s. We equally divide FMS into ten fuzzy match intervals such as: [0.9, 1.0), [0.8, 0.9) etc., and the index z specifies the corresponding interval. For example, since the fuzzy match score between s and tm_s in Figure 1 is 0.667, then $z = [0.6, 0.7)$.

Target Phrase Content Matching Status (TCM): It indicates the content matching status between $\overline{t}_k$ and $tm\_\overline{t}_{a(k)}$, and reflects the quality of $\overline{t}_k$. Because tm_t is nearly perfect when FMS is high, if the similarity between $\overline{t}_k$ and $tm\_\overline{t}_{a(k)}$ is high, it implies that the given $\overline{t}_k$ is possibly a good candidate. It is a member of {*Same, High, Low, NA* (*Not-Applicable*)}, and is specified as:

(1) If $tm\_\overline{t}_{a(k)}$ is not null:
   (a) if $FMS(\overline{t}_k, tm\_\overline{t}_{a(k)}) = 1.0$, $TCM_k = Same$;
   (b) else if $FMS(\overline{t}_k, tm\_\overline{t}_{a(k)}) > 0.5$, $TCM_k = High$;
   (c) else, $TCM_k = Low$;
(2) If $tm\_\overline{t}_{a(k)}$ is null, $TCM_k = NA$;

Here $tm\_\overline{t}_{a(k)}$ is null means that either there is no corresponding TM source phrase $tm\_\overline{s}_{a(k)}$ or there is no corresponding TM target phrase

$tm\_\bar{t}_{a(k)}$ aligned with $tm\_\bar{s}_{a(k)}$. In the example of Figure 1, assume that the given $\bar{s}_{a(k)}$ is "关联 $_5$ 的 $_6$ 对象 $_7$" and $\bar{t}_k$ is "*object that is associated*". If $tm\_\bar{t}_{a(k)}$ is "*object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", $TCM_k = Same$; if $tm\_\bar{t}_{a(k)}$ is "*an*$_1$ *object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", $TCM_k = High$.

Source Phrase Content Matching Status (SCM): Which indicates the content matching status between $\bar{s}_{a(k)}$ and $tm\_\bar{s}_{a(k)}$, and it affects the matching status of $\bar{t}_k$ and $tm\_\bar{t}_{a(k)}$ greatly. The more similar $\bar{s}_{a(k)}$ is to $tm\_\bar{s}_{a(k)}$, the more similar $\bar{t}_k$ is to $tm\_\bar{t}_{a(k)}$. It is a member of {*Same, High, Low, NA*} and is defined as:

(1) If $tm\_\bar{s}_{a(k)}$ is not null:
  (a) if $FMS(\bar{s}_{a(k)}, tm\_\bar{s}_{a(k)}) = 1.0$, $SCM_k = Same$;
  (b) else if $FMS(\bar{s}_{a(k)}, tm\_\bar{s}_{a(k)}) > 0.5$, $SCM_k = High$;
  (c) else, $SCM_k = Low$;
(2) If $tm\_\bar{s}_{a(k)}$ is null, $SCM_k = NA$;

Here $tm\_\bar{s}_{a(k)}$ is null means that there is no corresponding TM source phrase $tm\_\bar{s}_{a(k)}$ for the given source phrase $\bar{s}_{a(k)}$. Take the source phrase $\bar{s}_{a(k)}$ "关联 $_5$ 的 $_6$ 对象 $_7$" in Figure 1 for an example, since its corresponding $tm\_\bar{s}_{a(k)}$ is "关联 $_4$ 的 $_5$ 对象 $_6$", then $SCM_k = Same$.

Number of Linking Neighbors (NLN): Usually, the context of a source phrase would affect its target translation. The more similar the context are, the more likely that the translations are the same. Therefore, this NLN feature reflects the number of matched neighbors (words) and it is a vector of <*x*, *y*>. Where "*x*" denotes the number of matched source neighbors; and "*y*" denotes how many those neighbors are also linked to target words (not null), which also affects the TM target phrase selection. This feature is a member of {<*x*, *y*>: <2, 2>, <2, 1>, <2, 0>, <1, 1>, <1, 0>, <0, 0>}. For the source phrase "关联 $_5$ 的 $_6$ 对象 $_7$" in Figure 1, the corresponding TM source phrase is "关联 $_4$ 的 $_5$ 对象 $_6$" . As only their right neighbors "。 $_8$" and "。 $_7$" are matched, and "。 $_7$" is aligned with ".$_{10}$", NLN will be <1, 1>.

Source Phrase Length (SPL): Usually the longer the source phrase is, the more reliable the TM target phrase is. For example, the corresponding $tm\_\bar{t}_{a(k)}$ for the source phrase with 5 words would be more reliable than that with only one word. This feature denotes the number of words included in $\bar{s}_{a(k)}$, and is a member of {1, 2, 3, 4, ≥5}. For the case "关联 $_5$ 的 $_6$ 对象 $_7$", SPL will be 3.

Sentence End Punctuation Indicator (SEP): Which indicates whether the current phrase is a punctuation at the end of the sentence, and is a member of {*Yes*, *No*}. For example, the SEP for "关联 $_5$ 的 $_6$ 对象 $_7$" will be "*No*". It is introduced because the SCM and TCM for a sentence-end-punctuation are always "*Same*" regardless of other features. Therefore, it is used to distinguish this special case from other cases.

3.2 Model-II

As Model-I ignores the relationship among various possible TM target phrases, we add two features *TM Candidate Set Status* (CSS) and *Longest TM Candidate Indicator* (LTC) to incorporate this relationship among them. Since CSS is redundant after LTC is known, we thus ignore it for evaluating TCM probability in the following derivation:

$$
\begin{aligned}
&P(M_k|L_k, z) \\
&\triangleq P(TCM_k, LTC_k|SCM_k, NLN_k, CSS_k, SPL_k, SEP_k, z) \\
&\approx \left\{ \begin{matrix} P(TCM_k|SCM_k, NLN_k, LTC_k, SPL_k, SEP_k, z) \\ \times P(LTC_k|CSS_k, SCM_k, NLN_k, SEP_k, z) \end{matrix} \right\}
\end{aligned}
$$

The two new features CSS and LTC adopted in Model-II are defined as follows:

TM Candidate Set Status (CSS): Which restricts the possible status of $tm\_\bar{t}_{a(k)}$, and is a member of {*Single*, *Left-Ext*, *Right-Ext*, *Both-Ext*, *NA*}. Where "*Single*" means that there is only one $tm\_\bar{t}_{a(k)}$ candidate for the given source phrase $tm\_\bar{s}_{a(k)}$; "*Left-Ext*" means that there are multiple $tm\_\bar{t}_{a(k)}$ candidates, and all the candidates are generated by extending only the left boundary; "*Right-Ext*" means that there are multiple $tm\_\bar{t}_{a(k)}$ candidates, and all the candidates are generated by only extending to the right; "*Both-Ext*" means that there are multiple $tm\_\bar{t}_{a(k)}$ candidates, and the candidates are generated by extending to both sides; "*NA*" means that $tm\_\bar{t}_{a(k)}$ is null.

For "关联 $_4$ 的 $_5$ 对象 $_6$" in Figure 1, the linked TM target phrase is "*object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", and there are 5 other candidates by extending to both sides. Therefore, $CSS_k = Both\text{-}Ext$.

Longest TM Candidate Indicator (LTC): Which indicates whether the given $tm\_\bar{t}_{a(k)}$ is the longest candidate or not, and is a member of {*Original*, *Left-Longest*, *Right-Longest*, *Both-Longest*, *Medium*, *NA*}. Where "*Original*" means that the given $tm\_\bar{t}_{a(k)}$ is the one without extension; "*Left-Longest*" means that the given

$tm\_\bar{t}_{a(k)}$ is only extended to the left and is the longest one; "*Right-Longest*" means that the given $tm\_\bar{t}_{a(k)}$ is only extended to the right and is the longest one; "*Both-Longest*" means that the given $tm\_\bar{t}_{a(k)}$ is extended to both sides and is the longest one; "*Medium*" means that the given $tm\_\bar{t}_{a(k)}$ has been extended but not the longest one; "*NA*" means that $tm\_\bar{t}_{a(k)}$ is null.

For $tm\_\bar{t}_{a(k)}$ "*object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$" in Figure 1, $LTC_k = Original$; for $tm\_\bar{t}_{a(k)}$ "*an*$_1$ *object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", $LTC_k = Left\text{-}Longest$; for the longest $tm\_\bar{t}_{a(k)}$ "*an*$_1$ *object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$ *with*$_6$ *the*$_7$", $LTC_k = Both\text{-}Longest$.

3.3 Model-III

The abovementioned integrated models ignore the reordering information implied by TM. Therefore, we add a new feature *Target Phrase Adjacent Candidate Relative Position Matching Status* (CPM) into Model-II and Model-III is given as:

$$
\begin{aligned}
&P(M_k|L_k, z) \\
&\triangleq P([TCM, LTC, CPM]_k|[SCM, NLN, CSS, SPL, SEP]_k, z) \\
&\approx \left\{ \begin{array}{l} P(TCM_k|SCM_k, NLN_k, LTC_k, SPL_k, SEP_k, z) \\ \times P(LTC_k|CSS_k, SCM_k, NLN_k, SEP_k, z) \\ \times P(CPM_k|TCM_k, SCM_k, NLN_K, z) \end{array} \right\}
\end{aligned}
$$

We assume that CPM is independent with SPL and SEP, because the length of source phrase would not affect reordering too much and SEP is used to distinguish the sentence end punctuation with other phrases.

The new feature CPM adopted in Model-III is defined as:

Target Phrase Adjacent Candidate Relative Position Matching Status (CPM): Which indicates the matching status between the relative position of $[\bar{t}_{k-1}, \bar{t}_k]$ and the relative position of $[tm\_\bar{t}_{a(k-1)}, tm\_\bar{t}_{a(k)}]$. It checks if $[\bar{t}_{k-1}, \bar{t}_k]$ are positioned in the same order with $[tm\_\bar{t}_{a(k-1)}, tm\_\bar{t}_{a(k)}]$, and reflects the quality of ordering the given target candidate $\bar{t}_k$. It is a member of {*Adjacent-Same*, *Adjacent-Substitute*, *Linked-Interleaved*, *Linked-Cross*, *Linked-Reversed*, *Skip-Forward*, *Skip-Cross*, *Skip-Reversed*, *NA*}. Recall that $\bar{t}_k$ is always right adjacent to $\bar{t}_{k-1}$, then various cases are defined as follows:

(1) If both $tm\_\bar{t}_{a(k-1)}$ and $tm\_\bar{t}_{a(k)}$ are not null:
  (a) If $tm\_\bar{t}_{a(k)}$ is on the right of $tm\_\bar{t}_{a(k-1)}$ and they are also adjacent to each other:
    i. If the right boundary words of $\bar{t}_{k-1}$ and $tm\_\bar{t}_{a(k-1)}$ are the same, and the left boundary words of $\bar{t}_k$ and $tm\_\bar{t}_{a(k)}$ are the same, $CPM_k = Adjacent\text{-}Same$;
    ii. Otherwise, $CPM_k = Adjacent\text{-}Substitute$;
  (b) If $tm\_\bar{t}_{a(k)}$ is on the right of $tm\_\bar{t}_{a(k-1)}$ but they are not adjacent to each other, $CPM_k = Linked\text{-}Interleaved$;
  (c) If $tm\_\bar{t}_{a(k)}$ is not on the right of $tm\_\bar{t}_{a(k-1)}$:
    i. If there are cross parts between $tm\_\bar{t}_{a(k)}$ and $tm\_\bar{t}_{a(k-1)}$, $CPM_k = Linked\text{-}Cross$;
    ii. Otherwise, $CPM_k = Linked\text{-}Reversed$;

(2) If $tm\_\bar{t}_{a(k-1)}$ is null but $tm\_\bar{t}_{a(k)}$ is not null, then find the first $tm_\bar{t}_{a(k-n)}(k \geq n)$which is not null ($n$ starts from 2)[2]:
  (a) If $tm\_\bar{t}_{a(k)}$ is on the right of $tm\_\bar{t}_{a(k-n)}$, $CPM_k = Skip\text{-}Forward$;
  (b) If $tm\_\bar{t}_{a(k)}$ is not on the right of $tm\_\bar{t}_{a(k-n)}$:
    i. If there are cross parts between $tm\_\bar{t}_{a(k)}$ and $tm\_\bar{t}_{a(k-n)}$, $CPM_k = Skip\text{-}Cross$;
    ii. Otherwise, $CPM_k = Skip\text{-}Reversed$.

(3) If $tm\_\bar{t}_{a(k)}$ is null, $CPM_k = NA$.

In Figure 1, assume that $\bar{t}_{k-1}$, $\bar{t}_k$ and $tm\_\bar{t}_{a(k-1)}$ are "*gets an*", "*object that is associated with*" and "*gets*$_0$ *an*$_1$", respectively. For $tm\_\bar{t}_{a(k)}$ "*object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", because $tm\_\bar{t}_{a(k)}$ is on the right of $tm\_\bar{t}_{a(k-1)}$ and they are adjacent pair, and both boundary words ("*an*" and "*an*$_1$"; "*object*" and "*object*$_2$") are matched, $CPM_k = Adjacent\text{-}Same$; for $tm\_\bar{t}_{a(k)}$ "*an*$_1$ *object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", because there are cross parts "*an*$_1$" between $tm\_\bar{t}_{a(k)}$ and $tm\_\bar{t}_{a(k-1)}$, $CPM_k = Linked\text{-}Cross$. On the other hand, assume that $\bar{t}_{k-1}$, $\bar{t}_k$ and $tm\_\bar{t}_{a(k-1)}$ are "*gets*", "*object that is associated with*" and "*gets*$_0$", respectively. For $tm\_\bar{t}_{a(k)}$ "*an*$_1$ *object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", because $tm\_\bar{t}_{a(k)}$ and $tm\_\bar{t}_{a(k-1)}$ are adjacent pair, but the left boundary words of $\bar{t}_k$ and $tm\_\bar{t}_{a(k)}$ ("*object*" and "*an*$_1$") are not matched, $CPM_k = Adjacent\text{-}Substitute$; for $tm\_\bar{t}_{a(k)}$ "*object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", because $tm\_\bar{t}_{a(k)}$ is on the right of $tm\_\bar{t}_{a(k-1)}$ but they are not adjacent pair, therefore, $CPM_k = Linked\text{-}Interleaved$. One more example, assume that $\bar{t}_{k-1}$, $\bar{t}_k$ and $tm\_\bar{t}_{a(k-1)}$ are "*the annotation label*", "*object that is associated with*" and "*the*$_7$ *annotation*$_8$ *label*$_9$", respectively. For $tm\_\bar{t}_{a(k)}$ "*an*$_1$ *object*$_2$ *that*$_3$ *is*$_4$ *associated*$_5$", because $tm\_\bar{t}_{a(k)}$ is on the left of $tm\_\bar{t}_{a(k-1)}$, and there are no cross parts, $CPM_k = Linked\text{-}Reversed$.

[2] It can be identified by simply memorizing the index of nearest non-null $tm\_\bar{t}_{a(k-n)}$ during search.

4 Experiments

4.1 Experimental Setup

Our TM database consists of computer domain Chinese-English translation sentence-pairs, which contains about 267k sentence-pairs. The average length of Chinese sentences is 13.85 words and that of English sentences is 13.86 words. We randomly selected a development set and a test set, and then the remaining sentence pairs are for training set. The detailed corpus statistics are shown in Table 1. Furthermore, development set and test set are divided into various intervals according to their best fuzzy match scores. Corpus statistics for each interval in the test set are shown in Table 2.

For the phrase-based SMT system, we adopted the Moses toolkit (Koehn et al., 2007). The system configurations are as follows: GIZA++ (Och and Ney, 2003) is used to obtain the bidirectional word alignments. Afterwards, "intersection" [3] refinement (Koehn et al., 2003) is adopted to extract phrase-pairs. We use the SRI Language Model toolkit (Stolcke, 2002) to train a 5-gram model with modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998) on the target-side (English) training corpus. All the feature weights and the weight for each probability factor (3 factors for Model-III) are tuned on the development set with minimum-error-rate training (MERT) (Och, 2003). The maximum phrase length is set to 7 in our experiments.

In this work, the translation performance is measured with case-insensitive BLEU-4 score (Papineni et al., 2002) and TER score (Snover et al., 2006). Statistical significance test is conducted with re-sampling (1,000 times) approach (Koehn, 2004) in 95% confidence level.

4.2 Cross-Fold Translation

To estimate the probabilities of proposed models, the corresponding phrase segmentations for bilingual sentences are required. As we want to check what actually happened during decoding in the real situation, cross-fold translation is used to obtain the corresponding phrase segmentations. We first extract 95% of the bilingual sentences as a new training corpus to train a SMT system. Afterwards, we generate the corresponding phrase segmentations for the remaining 5% bilingual sentences with ***Forced Decoding*** (Li et al., 2000; Zollmann et al., 2008; Auli et al., 2009; Wisniewski et al., 2010), which searches the best phrase segmentation for the specified output. Having repeated the above steps 20 times[4], we obtain the corresponding phrase segmentations for the SMT training data (which will then be used to train the integrated models).

Due to OOV words and insertion words, not all given source sentences can generate the desired results through forced decoding. Fortunately, in our work, 71.7% of the training bilingual sentences can generate the corresponding target results. The remaining 28.3% of the sentence pairs are thus not adopted for generating training samples. Furthermore, more than 90% obtained source phrases are observed to be less than 5 words, which explains why five different quantization levels are adopted for Source Phrase Length (SPL) in section 3.1.

	Train	Develop	Test
#Sentences	261,906	2,569	2,576
#Chn. Words	3,623,516	38,585	38,648
#Chn. VOC.	43,112	3,287	3,460
#Eng. Words	3,627,028	38,329	38,510
#Eng. VOC.	44,221	3,993	4,046

Table 1: Corpus Statistics

Intervals	#Sentences	#Words	W/S
[0.9, 1.0)	269	4,468	16.6
[0.8, 0.9)	362	5,004	13.8
[0.7, 0.8)	290	4,046	14.0
[0.6, 0.7)	379	4,998	13.2
[0.5, 0.6)	472	6,073	12.9
[0.4, 0.5)	401	5,921	14.8
[0.3, 0.4)	305	5,499	18.0
(0.0, 0.3)	98	2,639	26.9
(0.0, 1.0)	**2,576**	**38,648**	**15.0**

Table 2: Corpus Statistics for Test-Set

4.3 Translation Results

After obtaining all the training samples via cross-fold translation, we use Factored Language Model toolkit (Kirchhoff et al., 2007) to estimate the probabilities of integrated models with Witten-Bell smoothing (Bell et al., 1990; Witten et al., 1991) and Back-off method. Afterwards, we incorporate the TM information $P(M_k|L_k, z)$ for each phrase at decoding. All experiments are

[3] "grow-diag-final" and "grow-diag-final-and" are also tested. However, "intersection" is the best option in our experiments, especially for those high fuzzy match intervals.

[4] This training process only took about 10 hours on our Ubuntu server (Intel 4-core Xeon 3.47GHz, 132 GB of RAM).

Intervals	TM	SMT	Model-I	Model-II	Model-III	Koehn-10	Ma-11	Ma-11-U
[0.9, 1.0)	81.31	81.38	85.44 *	86.47 *#	**89.41** *#	82.79	77.72	82.78
[0.8, 0.9)	73.25	76.16	79.97 *	80.89 *	**84.04** *#	79.74 *	73.00	77.66
[0.7, 0.8)	63.62	67.71	71.65 *	72.39 *	**74.73** *#	71.02 *	66.54	69.78
[0.6, 0.7)	43.64	54.56	54.88 #	55.88 *#	**57.53** *#	53.06	54.00	56.37
[0.5, 0.6)	27.37	46.32	47.32 *#	47.45 *#	**47.54** *#	39.31	46.06	47.73
[0.4, 0.5)	15.43	37.18	37.25 #	37.60 #	**38.18** *#	28.99	36.23	37.93
[0.3, 0.4)	8.24	29.27	**29.52** #	29.38 #	29.15 #	23.58	29.40	30.20
(0.0, 0.3)	4.13	**26.38**	25.61 #	25.32 #	25.57 #	18.56	26.30	26.92
(0.0, 1.0)	40.17	53.03	54.57 *#	55.10 *#	**56.51** *#	50.31	51.98	54.32

Table 3: Translation Results (BLEU%). Scores marked by "*" are significantly better ($p < 0.05$) than both TM and SMT systems, and those marked by "#" are significantly better ($p < 0.05$) than Koehn-10.

Intervals	TM	SMT	Model-I	Model-II	Model-III	Koehn-10	Ma-11	Ma-11-U
[0.9, 1.0)	9.79	13.01	9.22 #	8.52 *#	**6.77** *#	13.01	18.80	11.90
[0.8, 0.9)	16.21	16.07	13.12 *#	12.74 *#	**10.75** *#	15.27	20.60	14.74
[0.7, 0.8)	27.79	22.80	19.10 *#	18.58 *#	**17.11** *#	21.85	25.33	21.11
[0.6, 0.7)	46.40	33.38	32.63 #	32.27 *#	**29.96** *#	35.93	35.24	31.76
[0.5, 0.6)	62.59	39.56	**38.24** *#	38.77 *#	38.74 ***#**	47.37	40.24	38.01
[0.4, 0.5)	73.93	47.19	47.03 #	46.34 *#	**46.00** *#	56.84	48.74	46.10
[0.3, 0.4)	79.86	55.71	**55.38** #	55.44 #	55.87 #	64.55	55.93	54.15
(0.0, 0.3)	85.31	**61.76**	62.38 #	63.66 #	63.51 #	73.30	63.00	60.67
(0.0, 1.0)	50.51	35.88	34.34 *#	34.18 *#	**33.26** *#	40.75	38.10	34.49

Table 4: Translation Results (TER%). Scores marked by "*" are significantly better ($p < 0.05$) than both TM and SMT systems, and those marked by "#" are significantly better ($p < 0.05$) than Koehn-10.

conducted using the Moses phrase-based decoder (Koehn et al., 2007).

Table 3 and 4 give the translation results of TM, SMT, and three integrated models in the test set. In the tables, the best translation results (either in BLEU or TER) at each interval have been marked in bold. Scores marked by "*" are significantly better ($p < 0.05$) than both the TM and the SMT systems.

It can be seen that TM significantly exceeds SMT at the interval [0.9, 1.0) in TER score, which illustrates why professional translators prefer TM rather than SMT as their assistant tool. Compared with TM and SMT, Model-I is significantly better than the SMT system in either BLEU or TER when the fuzzy match score is above 0.7; Model-II significantly outperforms both the TM and the SMT systems in either BLEU or TER when the fuzzy match score is above 0.5; Model-III significantly exceeds both the TM and the SMT systems in either BLEU or TER when the fuzzy match score is above 0.4. All these improvements show that our integrated models have combined the strength of both TM and SMT.

However, the improvements from integrated models get less when the fuzzy match score decreases. For example, Model-III outperforms SMT 8.03 BLEU points at interval [0.9, 1.0), while the advantage is only 2.97 BLEU points at interval [0.6, 0.7). This is because lower fuzzy match score means that there are more unmatched parts between s and tm_s; the output of TM is thus less reliable.

Across all intervals (the last row in the table), Model-III not only achieves the best BLEU score (56.51), but also gets the best TER score (33.26). If intervals are evaluated separately, when the fuzzy match score is above 0.4, Model-III outperforms both Model-II and Model-I in either BLEU or TER. Model-II also exceeds Model-I in either BLEU or TER. The only exception is at interval [0.5, 0.6), in which Model-I achieves the best TER score. This might be due to that the optimization criterion for MERT is BLEU rather than TER in our work.

4.4 Comparison with Previous Work

In order to compare our proposed models with previous work, we re-implement two XML-Markup approaches: (Koehn and Senellart, 2010) and (Ma et al, 2011), which are denoted as **Koehn-10** and **Ma-11**, respectively. They are selected because they report superior performances in the literature. A brief description of them is as follows:

Source	如果 $_0$ 禁用 $_1$ 此 $_2$ 策略 $_3$ 设置 $_4$ ， $_5$ internet$_6$ explorer$_7$ 不 $_8$ 搜索 $_9$ internet$_{10}$ 查找 $_{11}$ 浏览器 $_{12}$ 的 $_{13}$ 新 $_{14}$ 版本 $_{15}$ ， $_{16}$ 因此 $_{17}$ 不 $_{18}$ 会 $_{19}$ 提示 $_{20}$ 用户 $_{21}$ 安装 $_{22}$ 。 $_{23}$
Reference	if$_0$ you$_1$ disable$_2$ this$_3$ policy$_4$ setting$_5$,$_6$ internet$_7$ explorer$_8$ does$_9$ not$_{10}$ check$_{11}$ the$_{12}$ internet$_{13}$ for$_{14}$ new$_{15}$ versions$_{16}$ of$_{17}$ the$_{18}$ browser$_{19}$,$_{20}$ so$_{21}$ does$_{22}$ not$_{23}$ prompt$_{24}$ users$_{25}$ to$_{26}$ install$_{27}$ them$_{28}$.$_{29}$
TM Source	如果 $_0$ 不 $_1$ 配置 $_2$ 此 $_3$ 策略 $_4$ 设置 $_5$ ， $_6$ internet$_7$ explorer$_8$ 不 $_9$ 搜索 $_{10}$ internet$_{11}$ 查找 $_{12}$ 浏览器 $_{13}$ 的 $_{14}$ 新 $_{15}$ 版本 $_{16}$ ， $_{17}$ 因此 $_{18}$ 不 $_{19}$ 会 $_{20}$ 提示 $_{21}$ 用户 $_{22}$ 安装 $_{23}$ 。 $_{24}$
TM Target	if$_0$ you$_1$ do$_2$ not$_3$ configure$_4$ this$_5$ policy$_6$ setting$_7$,$_8$ internet$_9$ explorer$_{10}$ does$_{11}$ not$_{12}$ check$_{13}$ the$_{14}$ internet$_{15}$ for$_{16}$ new$_{17}$ versions$_{18}$ of$_{19}$ the$_{20}$ browser$_{21}$,$_{22}$ so$_{23}$ does$_{24}$ not$_{25}$ prompt$_{26}$ users$_{27}$ to$_{28}$ install$_{29}$ them$_{30}$.$_{31}$
TM Alignment	0-0 1-3 2-4 3-5 4-6 5-7 6-8 7-9 8-10 9-11 11-15 13-21 14-19 15-17 16-18 17-22 18-23 19-24 21-26 22-27 23-29 24-31
SMT	if you disable this policy setting , internet explorer does not ***prompt users to install*** internet for new versions of the browser . [Miss 7 target words: 9~12, 20~21, 28; Has one wrong permutation]
Koehn-10	if you do you disable this policy setting , internet explorer does not check the internet for new versions of the browser , so does not prompt users to install them . [Insert two spurious target words]
Ma-11	if you disable this policy setting , internet explorer does not ***prompt users to install*** internet for new versions of the browser . [Miss 7 target words: 9~12, 20~21, 28; Has one wrong permutation]
Model-I	if you disable this policy setting , internet explorer does not ***prompt users to install*** new versions of the browser , so does not ***check the internet*** . [Miss 2 target words: 14, 28; Has one wrong permutation]
Model-II	if you disable this policy setting , internet explorer does not ***prompt users to install*** new versions of the browser , so does not ***check the internet*** . [Miss 2 target words: 14, 28; Has one wrong permutation]
Model-III	if you disable this policy setting , internet explorer does not check the internet for new versions of the browser , so does not prompt users to install them . [Exactly the same as the reference]

Figure 2: A Translation Example at Interval [0.9, 1.0] (with FMS=0.920)

Koehn et al. (2010) first find out the unmatched parts between the given source sentence and TM source sentence. Afterwards, for each *unmatched* phrase in the TM source sentence, they replace its corresponding translation in the *TM target sentence* by the corresponding *source phrase* in the input sentence, and then mark the substitution part. After replacing the corresponding translations of all unmatched source phrases in the TM target sentence, an XML input sentence (with mixed TM target phrases and marked input source phrases) is thus obtained. The SMT decoder then only translates the unmatched/marked source phrases and gets the desired results. Therefore, the *inserted* parts in the TM target sentence are automatically *included*. They use fuzzy match score to determine whether the current sentence should be marked or not; and their experiments show that this method is only effective when the fuzzy match score is above 0.8.

Ma et al. (2011) think fuzzy match score is not reliable and use a discriminative learning method to decide whether the current sentence should be marked or not. Another difference between Ma-11 and Koehn-10 is how the XML input is constructed. In constructing the XML input sentence, Ma-11 replaces each *matched* source phrase in the *given source sentence* with the corresponding TM target phrase. Therefore, the *inserted* parts in the TM target sentence are *not included*. In Ma's another paper (He et al., 2011), more linguistic features for discriminative learning are also added. In our work, we only re-implement the XML-Markup method used in (He et al., 2011; Ma et al, 2011), but do not implement the discriminative learning method. This is because the features adopted in their discriminative learning are complicated and difficult to re-implement. However, the proposed Model-III even outperforms the upper bound of their methods, which will be discussed later.

Table 3 and 4 give the translation results of Koehn-10 and Ma-11 (without the discriminator). Scores marked by "#" are significantly better ($p < 0.05$) than Koehn-10. Besides, the upper bound of (Ma et al, 2011) is also given in the tables, which is denoted as **Ma-11-U**. We calculate this

upper bound according to the method described in (Ma et al., 2011). Since He et al., (2011) only add more linguistic features to the discriminative learning method, the upper bound of (He et al., 2011) is still the same with (Ma et al., 2011); therefore, Ma-11-U applies for both cases.

It is observed that Model-III significantly exceeds Koehn-10 at all intervals. More importantly, the proposed models achieve much better TER score than the TM system does at interval [0.9, 1.0), but Koehn-10 does not even exceed the TM system at this interval. Furthermore, Model-III is much better than Ma-11-U at most intervals. Therefore, it can be concluded that the proposed models outperform the pipeline approaches significantly.

Figure 2 gives an example at interval [0.9, 1.0), which shows the difference among different system outputs. It can be seen that "*you do*" is redundant for Koehn-10, because they are insertions and thus are kept in the XML input. However, SMT system still inserts another "*you*", regardless of "*you do*" has already existed. This problem does not occur at Ma-11, but it misses some words and adopts one wrong permutation. Besides, Model-I selects more right words than SMT does but still puts them in wrong positions due to ignoring TM reordering information. In this example, Model-II obtains the same results with Model-I because it also lacks reordering information. Last, since Model-III considers both TM content and TM position information, it gives a perfect translation.

5 Conclusion and Future Work

Unlike the previous pipeline approaches, which directly merge TM phrases into the final translation result, we integrate TM information of each source phrase into the phrase-based SMT at decoding. In addition, all possible TM target phrases are kept and the proposed models select the best one during decoding via referring SMT information. Besides, the integrated model considers the probability information of both SMT and TM factors.

The experiments show that the proposed Model-III outperforms both the TM and the SMT systems significantly ($p < 0.05$) in either BLEU or TER when fuzzy match score is above 0.4. Compared with the pure SMT system, Model-III achieves overall 3.48 BLEU points improvement and 2.62 TER points reduction on a Chinese–English TM database. Furthermore, Model-III significantly exceeds all previous pipeline approaches. Similar improvements are also observed on the Hansards parts of LDC2004T08 (not shown in this paper due to space limitation). Since no language-dependent feature is adopted, the proposed approaches can be easily adapted for other language pairs.

Moreover, following the approaches of Koehn-10 and Ma-11 (to give a fair comparison), training data for SMT and TM are the same in the current experiments. However, the TM is expected to play an even more important role when the SMT training-set differs from the TM database, as additional phrase-pairs that are unseen in the SMT phrase table can be extracted from TM (which can then be dynamically added into the SMT phrase table at decoding time). Our another study has shown that the integrated model would be even more effective when the TM database and the SMT training data-set are from different corpora in the same domain (not shown in this paper). In addition, more source phrases can be matched if a set of high-FMS sentences, instead of only the sentence with the highest FMS, can be extracted and referred at the same time. And it could further raise the performance.

Last, some related approaches (Smith and Clark, 2009; Phillips, 2011) combine SMT and example-based machine translation (EBMT) (Nagao, 1984). It would be also interesting to compare our integrated approach with that of theirs.

Acknowledgments

The research work has been funded by the Hi-Tech Research and Development Program ("863" Program) of China under Grant No. 2011AA01A207, 2012AA011101, and 2012AA011102 and also supported by the Key Project of Knowledge Innovation Program of Chinese Academy of Sciences under Grant No.KGZD-EW-501.

The authors would like to thank the anonymous reviewers for their insightful comments and suggestions. Our sincere thanks are also extended to Dr. Yanjun Ma and Dr. Yifan He for their valuable discussions during this study.

References

Michael Auli, Adam Lopez, Hieu Hoang and Philipp Koehn, 2009. A systematic analysis of translation model search spaces. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 224–232.

Timothy C. Bell, J.G. Cleary and Ian H. Witten, 1990. Text compression: Prentice Hall, Englewood Cliffs, NJ.

Ergun Biçici and Marc Dymetman. 2008. Dynamic translation memory: using statistical machine translation to improve translation memory fuzzy matches. In *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2008)*, pages 454–465.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. *Technical Report TR-10-98*, Harvard University Center for Research in Computing Technology.

Yifan He, Yanjun Ma, Josef van Genabith and Andy Way, 2010. Bridging SMT and TM with translation recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 622–630.

Yifan He, Yanjun Ma, Andy Way and Josef van Genabith. 2011. Rich linguistic features for translation memory-inspired consistent translation. In *Proceedings of the Thirteenth Machine Translation Summit*, pages 456–463.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184.

Katrin Kirchhoff, Jeff A. Bilmes and Kevin Duh. 2007. Factored language models tutorial. *Technical report*, Department of Electrical Engineering, University of Washington, Seattle, Washington, USA.

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

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer and Ondřej Bojar. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 177–180.

Philipp Koehn, Franz Josef Och and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54.

Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *AMTA Workshop on MT Research and the Translation Industry*, pages 21–31.

Elina Lagoudaki. 2006. Translation memories survey 2006: Users' perceptions around tm use. In *Proceedings of the ASLIB International Conference Translating and the Computer 28*, pages 1–29.

Qi Li, Biing-Hwang Juang, Qiru Zhou, and Chin-Hui Lee. 2000. Automatic verbal information verification for user authentication. *IEEE transactions on speech and audio processing*, Vol. 8, No. 5, pages 1063–6676.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10 (8). pages 707–710.

Yanjun Ma, Yifan He, Andy Way and Josef van Genabith. 2011. Consistent translation using discriminative learning: a translation memory-inspired approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 1239–1248, Portland, Oregon.

Makoto Nagao, 1984. A framework of a mechanical translation between Japanese and English by analogy principle. In: *Banerji, Alick Elithorn and Ranan (ed). Artifiical and Human Intelligence: Edited Review Papers Presented at the International NATO Symposium on Artificial and Human Intelligence*. North-Holland, Amsterdam, 173–180.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29 (1). pages 19–51.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing 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)*, pages 311–318.

Aaron B. Phillips, 2011. Cunei: open-source machine translation with relevance-based models of each translation instance. *Machine Translation*, 25 (2). pages 166-177.

Richard Sikes. 2007, Fuzzy matching in theory and practice. *Multilingual*, 18(6):39–43.

Michel Simard and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, pages 120–127.

James Smith and Stephen Clark. 2009. EBMT for SMT: a new EBMT-SMT hybrid. In *Proceedings of the 3rd International Workshop on Example-*

Based Machine Translation (EBMT'09), pages 3–10, Dublin, Ireland.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas (AMTA-2006)*, pages 223–231.

Andreas Stolcke. 2002. SRILM-an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 311–318.

Guillaume Wisniewski, Alexandre Allauzen and François Yvon, 2010. Assessing phrase-based translation models with oracle decoding. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 933–943.

Ian H. Witten and Timothy C. Bell. 1991. The zero-frequency problem: estimating the probabilities of novel events in adaptive test compression. *IEEE Transactions on Information Theory*, 37(4): 1085–1094, July.

Ventsislav Zhechev and Josef van Genabith. 2010. Seeding statistical machine translation with translation memory output through tree-based structural alignment. In *Proceedings of the 4th Workshop on Syntax and Structure in Statistical Translation*, pages 43–51.

Andreas Zollmann, Ashish Venugopal, Franz Josef Och and Jay Ponte, 2008. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1145–1152.