Improving Translation Memory with Word Alignment Information

WU Hua, WANG Haifeng, LIU Zhanyi, TANG Kai
Toshiba (China) Research and Development Center
5/F., Tower W2, Oriental Plaza
No.1, East Chang An Ave., Dong Cheng District
Beijing, 100738, China
{wuhua, wanghaifeng, liuzhanyi, tangkai}@rdc.toshiba.com.cn

Abstract

This paper describes a generalized translation memory system, which takes advantage of sentence level matching, sub-sentential matching, and pattern-based machine translation technologies. All of the three techniques generate translation suggestions with the assistance of word alignment information. For the sentence level matching, the system generates the translation suggestion by modifying the translations of the most similar example with word alignment information. For sub-sentential matching, the system locates the translation fragments in several examples with word alignment information, and then generates the translation suggestion by combining these translation fragments. For pattern-based machine translation, the system first extracts translation patterns from examples using word alignment information and then generates translation suggestions with pattern matching. This system is compared with a traditional translation memory system without word alignment information in terms of translation efficiency and quality. Evaluation results indicate that our system improves the translation quality and saves about 20% translation time.

1 Introduction

The purpose of Translation Memory Systems (TMS) is to assist human translation by re-using pre-translated examples. A TMS consists of three parts: Translation Memory (TM), which records example translation pairs; a search engine, which retrieves the similar examples from the translation memory; and an on-line learning mechanism, which learns newly translated translation pairs. With such a system, any newly translated sentence, together with its translation, can be learned and added to the translation memory. Thus, the same sentence or text never needs to be translated more than once by humans. When translating a sentence, the TMS provides the translation of the best-matched example as the translation suggestion, which is sent to human translators for post-editing. Thus, TMS is also called a Machine Aided Human Translation System (MAHTS).

Generally speaking, TMS does not conduct real translation (Macklovitch & Russell, 2000). However, it can avoid repetitive labor and improve translation efficiency. It has been widely applied in technical document translation, product localization, etc. Now, a number of TMS tools are available at market such as Trados' Translator's WorkBench, SDL, IBM Translation Manager/2 and Transit.

In general, the traditional TMS retrieves examples matched with the input sentence at the sentence level (Planas and Furuse, 2000). It provides good translation suggestions only when there are closely matched examples in the TM. This leads to low coverage on unseen sentences or texts.

In order to solve this problem, many researchers use sub-sentential matching (Brown, 1996; Simard and Langlais, 2001; Huang et al., 2003). Brown (1996) segmented the input sentence into sequences of words and determined the translation of these sequences by performing sub-sentential alignment on each matched example. Simard and Langlais (2001) ranked the examples according to the length of matched sub-sequence of words. A longest available sub-sequence strategy was adopted to cover as much part of source sentences as possible. Huang et al. (2003) proposed a unified framework to generate translations with statistical confidences. Their experimental results indicated that sub-sentential matching improved translation efficiency and quality, which, in turn, saved the time of translators.

In this paper, we enhance the TMS by taking advantage of sentence level matching, sub-sentential matching, and pattern-based machine translation technologies. Sentence level matching provides good translation when there are closely matched examples in TM. Sub-sentential matching provides the translation suggestion by integrating

the translation fragments from several examples, which improves the coverage of the examples on unseen texts. The pattern-based machine translation technology makes inferences from the examples and provides the translation suggestions using pattern matching.

The above three technologies make use of word alignment information to generate translation suggestions. Sentence level matching uses word alignment information to replace the translations of the different parts in the example. Sub-sentential matching uses word alignment information to locate the translations of the matched parts in the examples. Pattern-based machine translation uses it to extract patterns.

Based on the above technologies, we implement a translation engine and a user interface. Our system is compared with a baseline system on English to Chinese translation. This baseline system also employs both sentence-level matching and sub-sentential level matching, but without word alignment information. Experimental results indicate that our system provides translation suggestions with higher accuracy and saves the translation time of the translators by about 20%.

The remainder of the paper is organized as follows. Section 2 describes the system architecture. Section 3 describes the sentence level matching technique, including similarity calculation and translation generation. Section 4 describes sub-sentential level matching, including sub-sequence combination and example selection. Section 5 describes translation pattern extraction and pattern combination for translation generation. Section 6 presents the evaluation results. The last section concludes this paper.

2 System Architecture

Our system employs three technologies: sentence level matching, sub-sentential matching, and pattern-based machine translation. First, the system searches the translation memory to find out whether there are closely matched examples, whose similarities with the input sentence are above a selected threshold. In this step, we use sentence level matching. If no closely matched example is found, the system segments the input sentence into several sub-sequences using the dynamic programming algorithm. And then it locates the corresponding translation part for each sub-sequence and combines them to form a translation suggestion. Otherwise, the system searches the pattern library to match possible patterns using the pattern-based translation technology. The detailed system flowchart is shown in Figure 1.

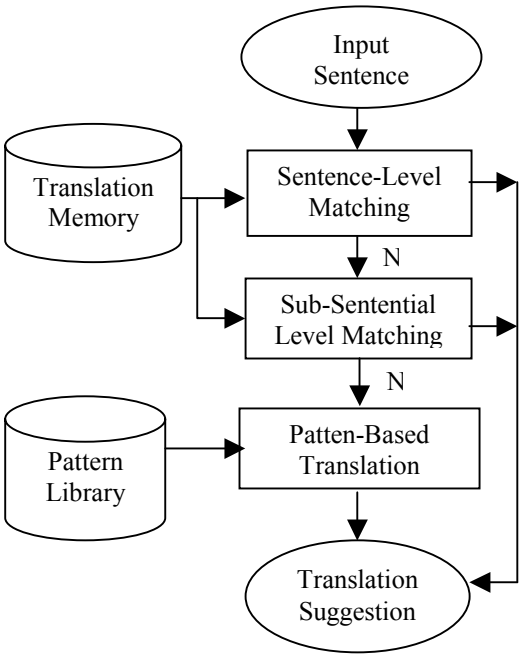

Figure 1. System Architecture

3 Sentence Level Matching

When translating a sentence, the TMS first searches the translation memory to find the closely matched sentences. The input sentence is compared with the source parts of the examples. Those examples whose similarities with the input sentence are above a threshold are presented to the users. If there is an exactly matched example, the target part of this example is suggested as translation. Otherwise, the target part of the example with the highest similarity score is modified and suggested as the translation.

Sentence-level matching is also used in traditional translation memory systems. The difference between our method and the previous methods lies in two aspects. One is the similarity calculation of the input sentence and the source parts of the examples. The other is the translation generation. The following two subsections address these two aspects.

3.1 Similarity Calculation

One widely used measure for similarity is a function of edit distance. However, the same value of edit distance of two examples with the input sentence sometimes does not mean the same semantic distance. For example:

Input sentence: We soon got books.
Example 1: We soon got flowers.
Example 2: We soon got talking.

The edit distances between the input sentence and the two examples are the same. However, the meaning between the input sentence and example 1 is closer than that between the input sentence and

example 2. Thus, except for calculating the edit distance, we also need to calculate the semantic distances of the different parts between the input sentence and examples. In addition, we also take into account the quality of the word alignment results of the examples in our TM. If the examples have poor alignment results, the quality of the translation suggestion based on these examples will also be bad.

Based on the above considerations, The similarity score between the input sentence and the source part of the example consists of three parts: a function of the edit distance, the semantic similarity of the different parts in the input sentence and the source part of the example, and the alignment confidence of the example.

Edit Distance:

The edit distance between two sentences $X = \{x_i \mid i > 0\}$ and $Y = \{y_j \mid j > 0\}$ is described as in equation (1).

$$D(i,j) = \begin{cases} 0 & \text{if i = 0, j = 0} \\ D(0, j-1) + tw(j) & \text{if i = 0, j} \neq 0 \\ D(i-1,0) + tw(i) & \text{if i} \neq 0\text{, j = 0} \\ \min \begin{pmatrix} D(i-1, j) + tw(i) \\ D(i, j-1) + tw(j) \\ D(i-1, j-1) + \Delta(i, j) \end{pmatrix} & \text{otherwise} \end{cases} \quad (1)$$

where

$$\Delta(i,j) = \begin{cases} 0 & \text{if } x_i = y_j \\ \infty & \text{otherwise} \end{cases}$$

$tw(k)\ (k = i, j)$ is the term weight. It is defined as follows:

$$tw(k) = \begin{cases} 0.5 & \text{if } fre(k) > \lambda_1 \\ 1 & \text{else} \end{cases}$$

$fre(k)$ is the occurring frequency of the term k in the translation memory.

Based on the edit distance, the first part of similarity between the input sentence and the source part of the example is shown in (2).

$$Sim_1(X,Y) = 1 - \frac{D(|X|, |Y|)}{\sum_{i=1}^{|X|} tw(i) + \sum_{j=1}^{|Y|} tw(j)} \quad (2)$$

Word Similarity:

For the different parts between the input sentence and the source part of the example, we calculate their semantic similarity using a thesaurus: WordNet (http://wordnet.princeton.edu/)[1]. The semantic similarity between two words is calculated as shown in (3), which is the same as described in (Lin, 1998).

$$WordSim(w_1, w_2) = \frac{2 * IC(LSuper(c_1, c_2))}{IC(c_1) + IC(c_2)} \quad (3)$$

where

c_1 is the concept subsuming the word w_1.

c_2 is the concept subsuming the word w_2.

$LSuper(c_1, c_2)$ is the lowest super-ordinate of c_1 and c_2.

$IC(c) = -\log p(c)$ is the information content of c. It is estimated using a semantically annotated corpus as shown in (4).

$$p(c) = \frac{\sum_{w \in W(c)} count(w)}{N} \quad (4)$$

$W(c)$ is the set of words in the corpus whose senses are subsumed by the concept c.

$count(w)$ is the occurring frequency of w.

N is the total number of word tokens in the corpus that also occur in the thesaurus.

Based on the word similarity, we calculate the similarity between the different parts in two sentences. If there are *m* and *n* different words in the input sentence and the source part of the example respectively, the similarity can be calculated as shown in (5).

$$Sim_2(X,Y) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} WordSim(w_i, w_j)}{m * n} \quad (5)$$

Alignment Confidence Score:

An example in the TM includes a source part and a target part. The words in these two parts are automatically aligned using a word aligner as described in (Wu and Wang, 2004). The quality of word alignment is evaluated using the alignment confidence score. The higher the score is, the better the quality is. The alignment score of an example Y is calculated as shown in (6).

$$AlignSocre(Y) = \exp\left(\frac{\sum_i \log(p(c_j \mid e_i))}{len(s)}\right) \quad (6)$$

where

$\exp(x) = e^x$;

[1] We take English to Chinese translation as a case study.

$p(c_j \mid e_i)$ is the probability of e_i aligned to c_j;

$len(s)$ is the number of words included in the source sentence s.

Final Similarity:

Combining the three factors described above, we calculate the similarity between the input sentence X and an example Y as shown in (7).

$$Sim(X,Y) = (Sim_1(X,Y) + (1 - Sim_1(X,Y)) * Sim_2(X,Y)^{\alpha}) * AlignScore(Y) \quad (7)$$

where α is a smoothing factor.

3.2 Translation Generation

With the method described in the above section, the system determines the most closely matched example, which has the highest similarity score with the input sentence. With the word alignment information, the system locates the translation of the different part in the most closely matched example and replaces the translation. The translation of the different parts in the input sentence is obtained using a bilingual dictionary with translation probability [2]. The modified translation is generated as the translation suggestion and is provided to the user for post-editing.

For example:

Input sentence: We soon got books.

Example: We soon got flowers.

我们不久得到了花。

The words underlined are the different part and its corresponding translation. When generating the translation, the system replaces the translation "花" of the word "flowers" with the Chinese word "书", which is the translation of the word "book". Thus, the translation suggestion provided to the user is "我们不久得到了书".

4 Sub-Sentential Matching

Sub-sentential matching enables the system to translate sentences on the sub-sentential level. This technology includes two parts. The first is to find optimal sub-sequences of the input sentence. The second is to find the translations of the sub-sequences using word alignment information.

4.1 Sub-Sequence Combination

For any sequence whose length is larger than two words in the input sentence, we search the translation memory to find whether there is any example matching with the sub-sequence. Thus, there are many combinations of the sub-sequences covering the input sentence. In order to find the optimal path of sub-sequences, we use dynamic programming algorithm. The algorithm is shown in Figure 2.

Input: the sentence to be translated $S = \{w_i \mid 1 \le i \le n\}$; The translation memory TM.
(1) For each sub-sequence s in S, search the TM to find whether it is included in TM. Let $Sub = \{s \mid s \in TM\}$ denotes the set of sub-sequences in the input sentences that are also found in TM. (2) For $i = 1$ *to* n For $j = i$ *to* n $\delta(i,j) = \max_l(\delta(i,j), \delta(i,l) + \delta(l,j))$ Where, $i < l < j$, $\delta(i,j)$ is the weight of the path from the word *i* to the word *j*.
Output: the path that maximizes $\delta(1,n)$

Figure 2. Dynamic Programming Algorithm

For any path from *i* to *j*, we consider the following three factors:

(1) The number of fragments matched: the fewer, the better.
(2) The length of each fragment: the longer, the better.
(3) The number of words uncovered by the TM. The fewer, the better.

Combing these factors, we calculate the weight of the path as shown in (8).

$$\delta(i,j) = c_1 * \frac{1}{\log(nFrag)} + c_2 * \frac{\sum_k LenFrag(k)}{nFrag} + c_3 * \log\frac{LenPath}{UnCover} \quad (8)$$

where

$nFrag$: the number of sub-sequences in the path that are found in the TM;

$LenFrag(k)$: the length of the k[th] sub-sequence in the path;

$LenPath$: the number of words in the path;

$UnCover$: the number of words that are not covered by any sub-sequence;

c_1, c_2, c_3 : the interpolation weights.

According to the dynamic programming algorithm, the system obtains the set of sub-

[2] This probability is trained from a word aligned bilingual corpus.

sequences in the input sentence that maximizes the path score $\delta(1, n)$.

4.2 Example Selection and Translation Generation

For each sub-sequence in the optimal path, there may be more than one example matching with it. This section describes how to select the best example for each sub-sequence. We consider two factors: the word alignment probability of the example and the length of the matched sub-sequence. Let $s_i \in OpSet$ ($OpSet$ is the set of sub-sequences in the optimal path). For each example containing the sub-sequence s_i, its weight is calculated as shown in (9). The example whose weight is the highest is selected for the sub-sequence s_i.

$$Wt_i(k) = \frac{LenSub(s_i)}{LenSrc(k)} * AlignScore(k) \quad (9)$$

where

$Wt_i(k)$ is the weight of the example k that containing the sub-sequence s_i;

$LenSub(s_i)$ is the length of the sub-sequence s_i;

$LenSrc(k)$ is the length of the source part of the example k;

$AlignScore(k)$ is the alignment score of the example k, which is calculated as shown in (6).

After obtaining the best example for each sub-sequence, we locate the translations in the examples using word alignment information and combine the translations to generate the translation suggestion. An example is shown in Figure 3.

The input sentence is segmented into two sub-sequences. The underlined words in the source part of the examples indicate the matched parts with the input sentence. And the underlined words in the target part describe the translations of the matched parts. The translation suggestion is generated by combining the translation fragments. The sub-sequences of the translation suggestion are not reordered according to structure of the target language. This is left to users for post-editing.

5 Pattern-Based Translation

Many example based machine translation systems perform translation based on translation patterns or templates extracted from bilingual corpora (Kaji et al., 1992; Güvenir and Cicekli, 1998; Carl, 1999; McTait, 2001). Kaji et al. (1992) used a bilingual dictionary and a parser to find phrase-level counterparts, based on which the translation

Input sentence:

All the data is displayed in the measurement display window only after the measurement data is fixed.

Examples:

(1) All the currently available data is displayed in the measurement display window during measurement.

测量 步骤 中 所有 当前 可 用 数据 都 显示 在 测量 显示 窗 中 。

(2) The traced area can be closed only after the measurement data is fixed.

只有 在 测量 数据 被 确定 后，所 描绘 的 区域 才会 闭合。

Translation Suggestion:

所有 数据 都 显示 在 测量 显示 窗 中， 只 有 在 测量 数据 被 确定 后。

Figure 3. An Example Showing Translation Generation

patterns were extracted. Carl (1999) used shallow parsing methods to extract patterns. Güvenir and Cicekli (1998) and McTait (2001) used language-neutral methods to extract translation patterns.

In this paper, we make use of the word alignment information and a source language parser to extract translation patterns from the examples. The extracted patterns are refined based on their occurring frequency. These patterns are then applied for sentence translation.

5.1 Translation Pattern Extraction

For examples in the TM, we first segment the sentences into words if necessary. Second, we align the words in the bilingual examples (Wu and Wang, 2004). Third, the source sentence in the examples are parsed (Amano et al., 1989). For noun phrases in the source sentence, we find their counterparts in the target sentence with word alignment information. If a noun phrase has a reliable counterpart in the target language, we use it and its counterpart as slots in the pattern.

To ensure high accuracy of the translation patterns, we refine them with their occurring frequency in the pattern library. If the pattern and its left part occurs N_i and N_i^l times respectively, the confidence score of this pattern is calculated as shown in (10). We only remain those patterns that occur more than a fixed threshold λ_2 and whose confidence score is larger than a threshold λ_3.

$$ConScore(p_i) = \frac{N_i}{N_i^l} \quad (10)$$

where p_i represents a translation pattern.

The detailed algorithm for translation pattern extraction is shown in Figure 4.

(1) Segment the sentences into words.
(2) Perform word alignment on the bilingual corpus.
(3) Parse the source sentences.
(4) For each noun phrase in the source sentence that has reliable translation, find its corresponding translation in the target sentence using word alignment results.
(5) Use noun phrases as slots to obtain translation patterns.
(6) Refine the translation patterns as shown in equation (10).

Figure 4. Translation Pattern Extraction

Figure 5 shows an example of the translation pattern. In the example, "he" and "that plan" are noun phrases. Their counterparts in the target sentences are "他" and "那个计划", respectively. These two phrases and their counterparts are extracted as slots. The brakets in the patterns display the slots and their numbers.

Example:

He gave that plan up.
他放弃了那个计划。

Alignment:

(He, 他) (gave…up, 放弃了) (that plan, 那个计划)

Translation Pattern:

(slot 1) gave (slot 2) up. → (slot 1) 放弃了 (slot 2) 。
he → 他
that plan → 那个计划

Figure 5. An Example of Translation Pattern

5.2 Pattern Combination

With the extracted translation patterns, the system translates the input sentence with pattern matching. Sometimes, the input sentence is translated by combining several patterns. The combination method is similar with that in (Güvenir and Cicekli, 1998). In this case, we calculate the confidence scores of the patterns used to generate the translation, which is described in (11).

$$PatternScore = \prod_i ConScore(p_i) \quad (11)$$

The pattern combination with the highest score is selected to generate the final translation. The detailed algorithm is described in Figure 6. The translations in the slots can be obtained with a bilingual dictionary or recursively generated with the patterns. Users can also modify the translations of the slots by referring to the bilingual dictionary.

(1) Use words in the input sentence to retrieve the matched patterns.
(2) Find the best matched patterns with equation (11).
(3) Fill out the slots and generate the translations.

Figure 6. Translation Generation

6 Evaluation

In this section, we evaluate the system in terms of translation efficiency and translation quality. Our system is compared with a traditional TMS (baseline system). This traditional TMS uses sub-sentential matching and sentential-level matching, but without word alignment information. Thus, the baseline system generates translation suggestion by providing the whole translation of the most similar example or the combinations of all translations of the examples containing the matched fragments. These two systems use the same user interface.

6.1 Experiment Setup

We evaluate the two systems by translating sentences in a specific domain (operating manual for a medical system). There are 5367 pre-translated English to Chinese sentence pairs in this domain. From them, we randomly select 400 sentence pairs, whose source parts are taken as testing data. The remaining data are word aligned and then indexed as translation examples.

We split the testing data into two sets. Each set includes 200 sentences. In order to ensure that the two testing sets are on the same level of difficulty, we evaluate the two testing sets with three measures: mean of sentence length, variance of sentence length, and the Flesch-Kincaid Grade Level (FKGL)[3]. The statistics of the two testing

[3] The Flesch-Kincaid Grade Level is a numerical indication of the grade level required to read and understand a given sample of text. It rates text based on the U.S. high school grade level system (i.e. a score of 7.0 would mean that a 7th grader should be able to comprehend the text).

sets are shown in Table 1. From the table, it can be seen that the two testing sets are on the same level of difficulty.

	Mean	Variance	FKGL
Testing Set 1	10.67	6.49	8.1
Testing Set 2	10.57	6.64	8.6

Table 1. Statistics of Two Testing Sets

Before evaluation, we use a held-out set including about 200 sentences to determine the thresholds described in the previous sections. With these thresholds, the translation quality (NIST score, described in section 6.3) of the held-out set is the highest.

6.2 Translation Efficiency

In the experiments, the users translate the sentences in the testing sets with the assistance of the two systems. We classify the human translators into two groups, with each group including three persons. Each person in the two groups translates all sentences in the two testing sets. But the translation order performed by these two groups is different.

Since the testing sets are selected from a specific domain, translators are not familiar with the terminology in the sentences. Thus, they may spend more time on the first system they used. In order to exclude this effect, we segment testing set 1 into three parts: 50 sentences ($Test_{11}$), 100 sentences ($Test_{12}$) and 50 sentences ($Test_{13}$). And we segment testing set 2 into two parts: 100 sentences ($Test_{21}$) and 100 sentences ($Test_{22}$). The testing order is shown in Table 2. For example, a translator in group one first translates $Test_{11}$ with the baseline system, and then translates $Test_{21}$ with our system, and so on.

	Group One	Group Two
$Test_{11}$	Baseline System	Our System
$Test_{21}$	Our System	Baseline System
$Test_{12}$	Baseline System	Our System
$Test_{22}$	Our System	Baseline System
$Test_{13}$	Baseline System	Our System

Table 2. Testing Order

When the users translate the sentences, the time that they spend on the translation is recorded. The evaluation results are shown in Table 3.

The numbers in Table 3 show the average minutes that the users spend on translations with different systems. The average time users spend with the baseline system (240 minutes) is more than that with our system (192 minutes). By using our system for translation, users can save about 20% of translation time[4].

	Baseline System	Our System
Group One	257	191
Group Two	223	193
Average	240	192

Table 3. Efficiency Evaluation Results

6.3 Translation Quality

Besides the translation efficiency, we also evaluate the quality of the translation suggestions produced by the two systems. Simard and Langlais (2001) evaluated the quality of the translation system in terms of precision and recall. In this paper, we also use precision and recall to evaluate the translation quality. We use S_G and S_R to represent the set of words in the translation suggestion provided by the system and the translation reference, respectively. The calculations of precision and recall are shown in (12) and (13).

$$precsion = \frac{|S_G \cap S_R|}{|S_G|} \quad (12)$$

$$recall = \frac{|S_G \cap S_R|}{|S_R|} \quad (13)$$

Besides precision and recall, we also use NIST score (Doddington, 2002) for evaluation. The NIST score is calculated by using the statistics of n-gram co-occurrence. It measures both the adequacy and fluency of the translation by comparing it with the translation references. Each sentence can have one or more translation references. It is reported that the NIST score correlates better with the human judgments than the BLEU score (Papineni et al., 2002). Although our system is not an automatic translation system, the adequacy and fluency measures of the translation suggestion are also useful for TMS. For each sentence in the testing sets, we have one translation reference.

The evaluation results are shown in Table 4. The results indicate that our system provides higher quality of translation suggestions than the baseline system.

	System	Precision	Recall	NIST
Test 1	Baseline	0.4011	0.4381	3.8405
Test 1	Ours	0.6485	0.5277	5.3385
Test 2	Baseline	0.4332	0.4791	4.0600
Test 2	Ours	0.6159	0.5329	5.5135

Table 4. Quality Evaluation Results

[4] The user interface is optimized for the baseline system. This interface is also used for our system without alternation. If we optimize it for our system, the translation efficiency may be further improved.

Besides the above evaluation, we also consider the contribution of the three technologies. Among 400 sentences in the two testing sets, 163 (40.75%) sentences are translated with sentence-level matching, 195 (48.75%) sentences are translated with sub-sentential matching, and 25 (6.25%) sentences are translated with patterns. And 17 sentences are not translated.

7 Conclusion

This paper describes a generalized translation memory system, which combines sentence level matching, sub-sentential level matching, and pattern-based machine translation technologies. All of the three technologies generate translation suggestions with the assistance of word alignment information in the examples. With sentence-level matching, our system generates translation suggestions by modifying the translations of the most similar examples. With the sub-sentential level matching and pattern-based translation, the system can also provide translation suggestions even if there are not very similar examples in the translation memory. The advantage of our system is that it combines the merits of the three technologies for translation generation and improves the coverage of the examples on unseen sentences or texts.

The system is compared with a traditional translation memory system in terms of translation efficiency and quality. Evaluation results indicate that our system improves the translation quality and saves about 20% of translation time.

References

ShinYa Amano, Hideki Hirakawa, Hiroyasu Nogami, and Akira Kumano. 1989. Toshiba Machine Translation System. *Future Computing Systems*, 2(3):227-246.

Ralf D. Brown. 1996. Example-Based Machine Translation in the Pangloss System. *In Proc. of the 16th International Conference on Computational Linguistics (COLING-1996)*, pages 169-174, Copenhagen, Denmark.

Michael Carl. 1999. Inducing Translation Templates for Example-Based Machine Translation. *In Proc. of the 7th Machine Translation Summit (MT Summit VII),* pages 250–258, Singapore.

George Doddington. 2002. Automatic Evaluation of Machine Translation Quality Using N-Gram Co-Occurrence Statistics. *In Proc. of the 2cd International Conference on Human Language Technology (HLT-2002)*, pages 138-145.

H. Altay Güvenir and Ilyas Cicekli. 1998. Learning Translation Templates from Examples. *Information Systems,* Vol. 23, No. 6, pages 353-363.

Jinxia Huang, Wei Wang, Ming Zhou. 2003. A Unified Statistical Model for Generalized Translation Memory System. *In Proc. of the 9th Machine Translation Summit (MT Summit IX)*, pages 173-180, New Orleans, Louisiana, USA.

Hiroyuki Kaji, Yuuko Kida, and Yasutsugu Morimoto. 1992. Learning Translation Templates from Bilingual Text. *In Proc. of the 14th International Conference on Computational Linguistics (COLING-1992),* pages 672-678, Nantes, France.

Dekang Lin. 1998. An Information-Theoretic Definition of Similarity. *In Proc. of International Conference on Machine Learning (ICML-1998)*, pages 296-304, Madison, Wisconsin.

Elliott Macklovitch and Graham Russell. 2000. What's been Forgotten in Translation Memory. *In Proc. of the 4th Conference of the Association for Machine Translation in the Americas (AMTA-2000),* pages 137-146.

Kevin McTait. 2001. Linguistic Knowledge and Complexity in an EBMT System Based on Translation Patterns. *Workshop on Example-Based Machine Translation, MT Summit VIII,* pages 23-24, Cuernavaca, Mexico.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. *In Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002),* pages 311-318, Philadelphia, PA, USA.

Emmanuel Planas and Osamu Furuse. 2000. Multi-level Similar Segment Matching Algorithm for Translation Memories and Example-Based Machine Translation. *In Proc. of the 18th International Conference on Computational Linguistics (COLING-2000),* pages 621-627, Saarbrücken, Germany.

Michel Simard and Philippe Langlais. 2001. Sub-Sentential Exploitation of Translation Memories. *In Proc. of the 8th Machine Translation Summit (MT Summit VIII),* pages 331-339, Santiago de Compostela, Galicia, Spain.

Hua Wu and Haifeng Wang. 2004. *Improving Domain-Specific Word Alignment with a General Bilingual Corpus.* In Robert E. Frederking and Kathryn B. Taylor (Eds.), Machine Translation: From Real Users to Research: 6th Conference of the Association for Machine Translation in the Americas (AMTA-2004), pages 262-271.