A Unified Statistical Model for Generalized Translation Memory System

Jin-Xia Huang[1]
CS Dept., Korea Advanced Institute of Scie. and Tech.
Daejeon, 305-701, Korea
hgh@world.kaist.ac.kr

Wei Wang[1]
New York University
NY 10003, USA
wei@cs.nyu.edu

Ming Zhou
Microsoft Research Asia
Sigma Centre, 49#, Zhichun Rd, Beijing, 100080, P.R.C
mingzhou@microsoft.com

Abstract

We introduced, for Translation Memory System, a statistical framework, which unifies the different phases in a Translation Memory System by letting them constrain each other, and enables Translation Memory System a statistical qualification. Compared to traditional Translation Memory Systems, our model operates at a fine grained sub-sentential level such that it improves the translation coverage. Compared with other approaches that exploit sub-sentential benefits, it unifies the processes of source string segmentation, best example selection, and translation generation by making them constrain each other via the statistical confidence of each step. We realized this framework into a prototype system. Compared with an existing product Translation Memory System, our system exhibits obviously better performance in the "assistant quality metric" and gains improvements in the range of 26.3% to 55.1% in the "translation efficiency metric".

1 Introduction

As a kind of Translation Aid tool, Translation Memory System (TMS) re-uses existing translations to perform its translation assistant task. A typical TMS consists of two parts: translation memory (TMEM), which records bilingual sentence pairs as examples; a search engine, which searches the most similar example (s) from TMEM. TMS provides the Target Language (TL) part of the best matched example to users for post-editing, and other matched examples could be provided to users as suggestions for translation. Normally, TMS does not conduct real translation (Macklovitch & Russell, 2000; Planas & Furuse, 2000).

TMS has been broadly used in technical document translation, localization, and other professional translation areas. These translation tasks have strict requirements on both translation quality and consistency. Not only the original text meaning, but also the presentation style should be rendered in target language (Boitet, 1994). For these requirements, TMS may not be simply replaced by fully-automatic Machine Translation system.

However, previous TMSs show obvious limitation in their translation assistant ability. They provide good translation references only when there are completely similar examples in TMEM, and therefore lead to low coverage on unseen texts (Macklovitch & Russell, 2000). It is because of their rudimentary matching technique that most of the previous TMSs do similarity calculation only on complete segments (Lepage, 1998; Planas & Furuse, 2000). The problem of the complete segment matching is that, an example normally tends to be excluded from the matching candidates even though it contains a useful sub-segment that could be helpful to the translation.

Making use of sub-sentence matching is significant to improve the translation coverage (Brown, 1996; Weale and Way, 1997; Macklovitch and Russell, 2000; Planas and Furuse, 2000; Simard and Langlais, 2001, Langlais and Simard, 2002).

[1] This work was finished while the authors were working at Microsoft Research Asia (MSRA).

Brown (1996, 1999) proposed a lexical Example-based Machine Translation approach in which no structural information was employed. His system segmented the input sentence into sequences of words occurring in the parallel corpus, and determined the translation of the word sequences by performing sub-sentential alignment on each matched example pair.

Simard and Langlais (2001) presented a Generalized TMS (GTMS), in which example matching phase was considered as an information retrieval process, so that sub-sentential sequences of words could be exploited from TMEM. Their search engine ranked examples with regard to the longest common sub-sequence of words. A longest available sub-sequences strategy was adopted in input sentence chunking and TL text generation to cover as much part of source sentence as possible. Langlais and Simard (2002) merged the Example-Based system with a statistical engine. The lattices of TL text were fed to a dynamic programming-based decoder to generate final translation.

Another irradiative work was Marcu (2001). He built a statistical TMEM to save bilingual word sequences, and produced the translation by using both TMEM and the statistical model so that the system could exploit the translation knowledge not only at word level but also at phrase level.

In this paper, we enhance the state-of-the-art of TMS by proposing a model to enable TMS a unified statistical qualification in the whole process of input sentence chunking, best example set selection, and the generation of a seamless translation without overlapping. We extend TMEM with three components: domain un-restricted monolingual corpora in both source language (SL) and TL; a bilingual Example Base (EB), which contains user-specified (and normally domain-restricted) bilingual sentence pairs and their word alignment information; a statistical Term Base (TB) which is automatically extracted from EB, contains bilingual word sequences (each word sequence includes at least one word either in SL or TL), their translation probabilities in the EB, and the information of their original examples which contain the word sequences in context. A confidence level prediction approach is proposed so that only dependable translations could be provided to user. A prototype system named EBMTLoc has been developed and a series of experiments show that, compared to a traditional TMS, LocStudio 4.5, an internal localization tool in Microsoft, EBMTLoc shows obviously better translation assistant quality especially when there is no very similar example in TMEM. The translation efficiency with the assistance of EBMTLoc shows remarkable improvement too, which is by the range of 26.3% to 55.5% in our experiments.

According to Simard and Langlais's (2001) definition on GTMS, that was, GTMS has ability to operate at a sub-sentential level while traditional TMS only handles complete sentence match, we call EBMTLoc a Statistical GTMS.

In the following, the model and EBMTLoc will be described in Section 2. The description of TMEM used in our system will be given in Section 3. The system evaluation and discussion will be shown in Section 4. Finally, the conclusions will be given in Section 5.

2 The Model and the System

Given an SL string $s_1^J = s_1 \ldots s_j \ldots s_J$, which is to be translated into a TL string $t_1^I = t_1 \ldots t_i \ldots t_I$; our translation procedure will be: 1) chunk s_1^J into a sequence of phrases $\tilde{s}_1^K$ (k=1,…,K; $\tilde{s}_k = s_{j_{k-1}+1}, \ldots, s_{j_k}$); 2) get the translation result $\tilde{t}_1^K$ ($\tilde{t}_k = t_{j_{k-1}+1}, \ldots, t_{j_k}$) through chunk-by-chunk translation; and 3) get the best translation result $\hat{t}_1^I$ by a target language model. Denote the SL and TL monolingual corpora with C_S and C_T, the bilingual EB with E, we have,

$$
\begin{aligned}
\hat{t}_1^I &= \operatorname*{argmax}_{t_1^I}\{\Pr(t_1^I \mid s_1^J)\} \\
&= \operatorname*{argmax}_{t_1^I}\{\Pr(\tilde{s}_1^K \mid s_1^J, C_S, E) \cdot \Pr(\tilde{t}_1^K \mid \tilde{s}_1^K, s_1^J, E) \cdot \Pr(t_1^I \mid \tilde{t}_1^K, \tilde{s}_1^K, s_1^J, C_T)\} \\
&\approx \operatorname*{argmax}_{t_1^I}\{\underbrace{\Pr(\tilde{s}_1^K \mid s_1^J, C_S, E)}_{\text{chunking model}} \cdot \underbrace{\Pr(\tilde{t}_1^K \mid \tilde{s}_1^K, E)}_{\text{translation model}} \cdot \underbrace{\Pr(t_1^I \mid \tilde{t}_1^K, C_T)}_{\text{language model}}\}
\end{aligned}
$$

Here, we use a direct translation model instead of source-channel approach (Och and Ney, 2002). Considering the model is for a TMS, and what we want is not only the best TL translation $\hat{t}_1^I$, but also the best example set $\hat{e}$ to support the translation $\hat{t}_1^I$, we have,

$$< \hat{t}_1^I, \hat{e} > \approx \arg\max_{t_1^I, e} \{ \underbrace{\Pr(e', \tilde{s}_1^K \mid s_1^J, C_S, E)}_{\text{source language chunking model}} \cdot \underbrace{\Pr(e, \tilde{t}_1^K \mid \tilde{s}_1^K, e')}_{\text{translation model}} \cdot \underbrace{\Pr(t_1^I \mid \tilde{t}_1^K, C_T)}_{\text{target language model}} \} \quad (1)$$

Where e' is the example set which is related to $\tilde{s}_1^K$ ("related" means each example in e' contains at least one chunk $\tilde{s}_k$ of $\tilde{s}_1^K$). e is a sub-set of e' which is related to both $\tilde{t}_1^K$ and $\tilde{s}_1^K$.

In practice, we use phrase based unigram in language model to avoid data sparseness problem.

2.1 Chunking

The task of SL chunking model is, given an input string s_1^J, a SL monolingual corpus C_S and a bilingual EB E, get the SL chunking result $\tilde{s}_1^K$.

A **chunk** (phrase) is a continuous of base chunks with non-zero word in it. A **base chunk** is equal to a syntactic constituent in CoNLL-2000 (Erik et al., 2000). And if several continuous base chunks in input sentence are continuous too in any one example, then they will have opportunity to be combined to a "chunk (phrase)".

There are two sub-phases in chunking phase, the first one is monolingual corpus based chunking to get base chunk $\tilde{s}'^{K'}_1$; the second one is example based chunk combination phase to get final chunk $\tilde{s}_1^K$, in which the chunk is normally as long as better to get more natural translation. Consider the system gets the example set e' from E after the chunking procedure, we could omit it from the equation. We have,

$$\begin{aligned}
\hat{\tilde{s}}_1^K &= \arg\max_{\tilde{s}_1^K} \{\Pr(\tilde{s}_1^K \mid s_1^J, C_S, E)\} \\
&= \arg\max_{\tilde{s}_1^K} \{\sum_{\tilde{s}'^{K'}_1} \Pr(\tilde{s}_1^K, \tilde{s}'^{K'}_1 \mid s_1^J, C_S, E)\}, \quad \text{(hidden variable } \tilde{s}'^{K'}_1 \text{ is introduced)} \\
&\approx \arg\max_{\tilde{s}_1^K} \{\max_{\tilde{s}'^{K'}_1} \Pr(\tilde{s}_1^K, \tilde{s}'^{K'}_1 \mid s_1^J, C_S, E)\}, \quad \text{(max approximation is introduced)} \\
&\approx \arg\max_{\tilde{s}_1^K} \{\max_{\tilde{s}'^{K'}_1} [\Pr(\tilde{s}'^{K'}_1 \mid s_1^J, C_S) \cdot \Pr(\tilde{s}_1^K \mid \tilde{s}'^{K'}_1, E)]\}
\end{aligned}$$

Introduce the independent approximation, get,

$$\hat{\tilde{s}}_1^K \approx \underbrace{\max_{\tilde{s}'^{K'}_1} \Pr(\tilde{s}'^{K'}_1 \mid s_1^J, C_S)}_{\text{base chunking}} \cdot \underbrace{\arg\max_{\tilde{s}_1^K} \{\Pr(\tilde{s}_1^K \mid \tilde{s}'^{K'}_1, E)\}}_{\text{example based chunk combination}} \quad (2)$$

In base chunking, we follow the SL monolingual chunking model that is introduced by Wang et al. (2002), in which 13 chunk types were used for English, which is the same with Erik et al. (2000). In experiment, 92.52% of precision and 90.81% of recall in English base chunking is obtained, which is above the average level in CoNLL-2000 (Erik et al., 2000).

The second sub-phase is example-based chunk combination. If two or more than two continuous base chunks appear in any example, they have the opportunity to be combined to one chunk.

We express the procedure of getting $\Pr(\tilde{s}_1^K \mid \tilde{s}'^{K'}_1, E)$ with function $F(\tilde{s}'^{K'}_1 \to \tilde{s}_1^K \mid E)$. Consider the data sparseness problem, submit to losing some of linguistic constraints, we get a back off function $L(\tilde{s}'^{K'}_1 \to \tilde{s}_1^K \mid E)$ which is based on a weighted length to substitute the function F. We have:

$$\Pr(\tilde{s}_1^K \mid \tilde{s}'^{K'}_1, E) \approx L(\tilde{s}'^{K'}_1 \to \tilde{s}_1^K \mid E)$$

Here is an example for chunk combination. Suppose $\tilde{s}'^{K'}_1 = \tilde{s}'_1 \tilde{s}'_2 \tilde{s}'_3 .. \tilde{s}'_{K'}$ after base chunking, and the chunk $\tilde{s}'_1$ and $\tilde{s}'_2$ appear in one example as continuous chunks, while $\tilde{s}'_2$ and $\tilde{s}'_3$ appear and be continuous in another example, then we could have:

$$\tilde{s}_1^K = \begin{cases} [\tilde{s}'_1 \ \tilde{s}'_2][\tilde{s}'_3]...[... \ \tilde{s}'_{K'}] \\ [\tilde{s}'_1][\tilde{s}'_2 \ \tilde{s}'_3]...[... \ \tilde{s}'_{K'}] \end{cases}$$

2.2 Example-based Phrase Translation Model

The translation model is different from existing ones in two respects: first, it is a translation model on phrase (including word) level and all phrases are related to their original examples. Second, the model should give the TL translations with the original examples which support the translations.

Suppose the input is a phrase sequence $\tilde{s}_1^K$ and its related example set e', then the translation procedure is: get related examples $\{e_{\tilde{s}_k}\}$ to each phrase $\tilde{s}_k$; get TL correspondences $\{\tilde{t}_k\}$ of $\tilde{s}_k$ from $\{e_{\tilde{s}_k}\}$; get the probability of the translation of each $\tilde{t}_1^K$ and its example set e with equation (3). The best translation $\hat{\tilde{t}}_1^K$ of $\tilde{s}_1^K$ and the best example set $\hat{e}$ for the translation should gain the highest Pr(e, $\tilde{t}_1^K \mid \tilde{s}_1^K$, e').

$$\Pr(e, \tilde{t}_1^K \mid \tilde{s}_1^K, e') = \prod_k \underbrace{\Pr(e_{\tilde{s}_k} \mid \tilde{s}_k, e')}_{\text{example selection model}} \cdot \underbrace{\Pr(\tilde{t}_k \mid \tilde{s}_k, e_{\tilde{s}_k})}_{\text{translation model}} \quad (3)$$

There are an example selection model and a phrase-based translation model in the example-based translation model. The distortion model is not adopted because our model is for a TMS and only partial translation will be provided to users.

Example selection model Given a SL phrase sequence $\tilde{s}_1^K = \tilde{s}_1 ... \tilde{s}_k ... \tilde{s}_K$, the example selection model selects the matching example $e_{\tilde{s}_k}$ to each phrase $\tilde{s}_k$ for any $1 \le k \le K$. Assume that the more similar example would be more helpful to the translation of the input string, the model could be simplified to a similarity calculation:

$$
\begin{aligned}
&\Pr(e_{\tilde{s}_k} \mid \tilde{s}_k , e') \\
&\approx Sim\,(\tilde{s}_1^K , e_{\tilde{s}_k}), (\text{where } e_{\tilde{s}_k} \in e') \\
&= \frac{1}{| e_{\tilde{s}_k} |} \sum_{i=1}^{K} | \tilde{s}_i |_{(\tilde{s}_i \subset \tilde{s}_1^K , e_{\tilde{s}_k})}
\end{aligned} \tag{4}
$$

In above equation, $|x|$ means the weighted length (word number) of string x. A content word gains 1.0, where it is 0.4 and 0.2 for a function word and a character in our experiment.

The translation model Given a SL phrase $\tilde{s}_k$ and the matching example $e_{\tilde{s}_k}$, what we want to get is the TL translation $\tilde{t}_k$.

Consider data sparseness problem may result in the undependable corpus-dependent translation probability $\Pr(\tilde{t}_k \mid \tilde{s}_k , E)$ (simply $\Pr(\tilde{t}_k \mid \tilde{s}_k)$) which is estimated from the EB, we introduce example-dependent translation likelihood of $\tilde{s}_k$ and $\tilde{t}_k$ in given matching example $e_{\tilde{s}_k}$, which is denoted by $SF(\tilde{t}_k \mid \tilde{s}_k , e_{\tilde{s}_k})$. We have:

$$\Pr(\tilde{t}_k \mid \tilde{s}_k, e_{\tilde{s}_k}) \approx \alpha \times \Pr(\tilde{t}_k \mid \tilde{s}_k) + (1-\alpha) \times \beta \times SF(\tilde{t}_k \mid \tilde{s}_k, e_{\tilde{s}_k}) \tag{5}$$

where, α is corpus-dependent coefficients, and β is interpolation coefficient for using translation likelihood with translation probability. Intuitively, α should be low and so the example-based translation likelihood $SF(\tilde{t}_k \mid \tilde{s}_k, e_{\tilde{s}_k})$ could be higher weighted when $\Pr(\tilde{t}_k \mid \tilde{s}_k)$ is not informative (for example, $\tilde{s}_k$ could be translated to five TL phrases, and each translation probability between them is 20%), and α could be high otherwise. In our experiments, α is set to 1.0 when the co-occurrence frequency of $\tilde{s}_k$ and $\tilde{t}_k$ in EB is larger than 5, and the corpus-dependent translation probability $\Pr(\tilde{t}_k \mid \tilde{s}_k)$ is higher than 0.4; Otherwise α will be set to 0.

The corpus-dependent translation probability is estimated from the bilingual EB and saved to TB in advance to raise the performance; and the example-dependent translation likelihood is gotten through phrase alignment, which we describe in TMEM building (Chapter 3).

2.3 Decoding Algorithm

Consider the decoding as a best path finding problem, a dynamic programming algorithm is employed in our system (figure 1). The proposed integrated model (equation 1), in which the distortion model and language model is omitted for the specific object TMS, is employed as the evaluation function in the algorithm.

```
Input: W, the weight matrix for a path with vertices v1, …, vn;
and n.
Output: D, an n×n matrix; D[i, j] = distance from vi to vj.
  Procedure Distances (W: Matrix; n:integer; D: Matrix);
  Begin
        D := W;
        for i := 1 to n do
        for j := i to n do
        for k := i to j do
            D[i, j] := min(D[i, j], D[i, k]+D[k, j])
        end end end {for loops}
   end {Distances}
```

Figure 1. Dynamic-Programming Algorithm

Here is the implementation strategy of the algorithm:

- Select n-best chunking results by equation (2)
- Select n-best examples for each chunk by equation (4).
- Edge each chunk with related example ID and the similarity weight of the example that is acquired from equation (4). For example, if there are two related examples to the same chunk, each with ID 12007, 52773 and similarity weight 0.31, 0.77, respectively, then there will be two edges given to the chunk, and each of them is marked with <12007, 0.31> and <52773, 0.77>, respectively.

Figure 2. Find the best path

- Get translation probability of each edge from equation (5), multiply by the edge weight (example similarity weight), and write the reciprocal of the weight into the weight matrix *W*.
- Get each path weight and finally the best TL translation and the best example set $<\hat{t}_1^J,\hat{e}>=<\hat{\tilde{t}}_1^K,\hat{e}>$ by the algorithm figure 1.

2.4 Confidence Level Prediction

Most TMSs provide confidence levels. Confidence level is helpful to user for reference in post-edition, as well as it can be used to cut off undependable translation outputs that system does not confident enough. However, according to the translation methodology, the confidence levels of existing TMSs are only related to the similarity between input string and the employed example.

As a GTMS, The confidence level is in inverse proportion to the number of the adopted examples and the chunk number of the input string, and in direct proportion to the final translation percentage and the alignment confidence of each translated chunk. Alignment confidence level is related to the consistence between aligned words and the unaligned word number in the phrase alignment result. The confidence level could be expressed with equation (6).

$$\begin{aligned} ConL = & c_1 \times f_1(|\hat{e}|^{-1}) + c_2 \times f_2(K^{-1}) \\ & + c_3 \times f_3(AlignConL) + c_4 \times f_4(TransPercent) \end{aligned} \quad (6)$$

Where, *ConL* denotes the translation confidence level; $|\hat{e}|$ is the employed example number; *K* is the chunk number of the input string; *AlignConL* is average alignment confidence level of each chunk; *TransPercent* is weighted translation percentage of the input string.

3 TMEM Building

There are three components in the TMEM. One is domain unrestricted monolingual corpora; one is bilingual EB with word alignment information; and the final one is bilingual TB with translation probability. Monolingual corpora will be used in SL chunking and TL generation, while the EB and TB will be employed in the whole procedure of input sentence chunking, best example set selection and translation generation. The word alignment information and TB are automatically acquired. In this chapter, we give description on how we build the EB and TB.

Example Base EB saves bilingual sentence pairs with their word alignment information. An inverted example bi-term index is saved in a table to speed up the example retrieval process (Table 1). The second column shows the related examples' ID and the position of the bi-term in each example. For example, bi-term "variation over" occurs in example "771" with position 13 and example "776" with position 4 and 27.

value type	28948|6
variation over	771|13_776|4^27

Table 1. Bi-term indexing for EB

The bilingual examples are saved in another table with their IDs and word alignment results (Table 2). From table 2, we can see the example ID is "28948", the SL and TL parts have been segmented, and the word alignment and the alignment confidence level which reflect the consistence between two TL and SL words have been given after the "[A]" mark.

(ID: 28948); ([SL] Operand/1 is/2 something/3 other/4 than/5 a/6 value/7 type/8 ./9); ([TL] 操作数/1 不是/2 数值/3 类型/4 。/5); ([A] (1:1_0.32); (7:3_0.64); (8:4_0.64); (9:5_0.64).

Table 2. Bilingual example with word alignment result

The word alignment results are acquired automatically by using following the approach introduced in Wang et al. (2001). The precision of the word alignment is 88.48%, and the recall is 76.36% in software domain.

Term Base TB is automatically exacted from EB, it is saved into TMEM for raising the performance. TB saves bilingual word sequences (terms) and their corpus-dependent translation probabilities $\Pr(\tilde{t}\mid\tilde{s})$ (Table 3).

SL ($\tilde{s}$)	TL ($\tilde{t}$)	Freq	$\Pr(\tilde{t}\mid\tilde{s})$
Action	操作^动作^Action	354^96^59	0.66^0.18^0.11
action menu	“动作”菜单	24	0.96

Table 3. Term-Base (corpus-dependent translation probability)

From the table 3, we can see the SL term "action" has three translations in TL: "操作

([*caozuo*], means 'operation'), " 动作 （[*dongzuo*], means 'movement'）, "Action (the same with its original form in English, for menu item)", each with frequency "354", "96" and "59" in the EB and translation probability "0.66", "0.18" and "0.11", respectively.

There are two steps in TB extraction. Firstly chunk the bilingual EB by following the bilingual chunking approach introduced in Wang et al. (2002) to get syntactic chunks, and then perform phrase alignment on all of the SL chunk sequences by following the phrase alignment approach introduced in Wang et al. (2001).

In practice, the phrase pair recorded to TB should satisfy requirements of: $\tilde{s}$ is not longer than 5 words; the co-occurrence of $\tilde{s}$ and $\tilde{t}$ ("Freq" in table 3) is higher than 5 (times); and $\Pr(\tilde{t} \mid \tilde{s})$ is higher than 0.10. In the experiment, we extract 14,795 phrase items from 62,844 example pairs with 90% of translation precision in software domain.

4 Experiment

To give the meaningful evaluation result, the experiment should answer the questions from users that, how much improvement in the quality and the efficiency could be expected compare to the traditional TMS? To give as fair answer as possible, a real localization task is simulated in our experiment.

4.1 Experiment Design

The EBMTLoc engine is evaluated from two aspects: "translation efficiency" and "translation assistant quality". The traditional TMS employed for comparison is LocStudio 4.5, which is an internal tool of Microsoft for software localization. It is a typical TMS that employs full segment matching methodology like most of the existing TMSs.

The experiment environments, including the test set and the EB of the TMEM are the same to the two systems. The EB contains the same 62,844 string pairs that from Office 2000 software, and the two test set contains totally 400 strings that are randomly from Office XP software. The precision and the recall of the EBMTLoc EB word alignment are 88.48% and 76.36%, respectively, while the TB of EBMTLoc contains 14,795 items with 90% of precision (see chapter 2).

The two right most columns in table 4 show the conditions of the test set. The "matching ratio" is from the "confidence level" of LocStudio, which reflects the similarity between the inputs and the most similar examples of them in TMEM.

Matching Ratio		Count	Per	Assistant Quality: T-TMS better	E-Loc better	Both good	Both Bad
==100		207	52%	10	3	192	2
>= 90 and <100		10	2%			10	
>=80 and <90		14	4%		9	5	
>=60 and <80		56	14%		45	2	9
>=40 and <60		57	14%		46	1	10
>=25 and <40		28	7%		15	2	11
<25		28	7%		9	2	17
Total	Count	400		10	127	214	49
	Percent		100%	2.5%	31.75%	53.5%	12.25%

Table 4. Test set and the translation assistant quality

4.2 Translation Assistant Quality

There are two ways in the evaluation of translation assistant quality. The first one is **relative evaluation.** Four evaluation groups (one person per group) are required to compare the translations from the two systems, and mark each test string with "T-TMS is better", "E-Loc is better", "both good" and "both bad" (See column "Assistant Quality" in table 4. "T-TMS" is for LocStudio 4.5 means "traditional TMS", "E-Loc" is for EBMTLoc).

From the table 4, we can see that EBMTLoc shows obviously better result than traditional TMS. Especially when consider only fuzzy matching cases, in which matching ratio do not reach 100%, 64.2% of the translations are marked with "E-Loc is better", while the cases of "T-TMS is better" is only 0.5%.

The second way in assistant quality evaluation is **absolute evaluation.** All of the suggestions and the translations from the two systems will be valued by the four evaluation groups with score "-1, 0, 1, 2, 3, 4" independently (-1 is for "bothersome"; 0 is "no help"; 1 is "little help"; 2 is "helpful"; 3 is "very helpful"; and 4 is for "perfect", respectively).

Figure 3 shows the average scores of the two systems. The suggestions and translations from EBMTLoc and LocStudio gain 3.44, 3.28, 2.81 and 2.18, respectively. Both of the results of EBMTLoc (3.44 and 3.28) are in the interval between "very helpful" and "perfect"; while the results of the LocStudio (2.81 and 2.18) are in the interval between "helpful" and "very helpful". If only consider the fuzzy matching cases, the average scores are 2.84, 2.52, 1.54, and 0.25 respectively.

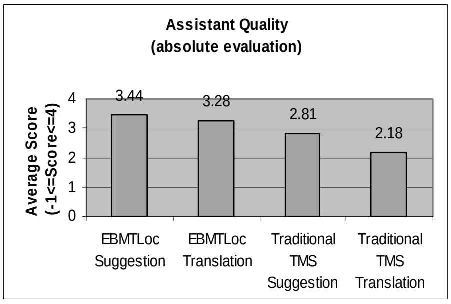

*Figure 3.*Absolute evaluation in assistant quality

From both of the absolute evaluation and relative evaluation, we can see that the translation assistant quality of EBMTLoc is significantly better than that of traditional TMS, and such tendency is more obvious when consider only the fuzzy matching cases, which should be the main target of the TMS.

4.3 Translation Efficiency

In "translation efficiency evaluation", each evaluation group will be required to translate (localize) given text under the help of EBMTLoc system first, and then re-translate the re-ordered text under the help of traditional TMS, or vice versa. We assume that, the translators (evaluation groups) are "new hands" when they translate the given text at their first time; and when they re-translate it (with another tool), they have become "skilled translators", because at this time they become more familiar with the input text and the translation.

Table 5 shows the experiment design and the results. The second column "Eva." shows the testing group; the third column shows the test set, in which each test set contains 200 sentences randomly from Office XP software; the column "Time (min)" shows the working time for translation. The test with index "5" and "6" are performed additionally for confirmation after the test "1" to "4" show obvious tendency that EBMTLoc shows better efficiency than traditional TMS.

Index	Eva.	Test Set	1st run	Time (min)	2nd run	Time (min)
1	G1	T1	E-Loc	60	T-TMS	80
2	G2	T2	E-Loc	60	T-TMS	80
3	G3	T1	T-TMS	116	E-Loc	50
4	G4	T2	T-TMS	165	E-Loc	66
5	G2	T1	T-TMS	115	E-Loc	50
6	G3	T2	E-Loc	58	T-TMS	65

Table 5. Efficiency evaluation

From the table 5, we can see that, the average translation time with EBMTLoc is 59.3 minutes for "new hands" and 55.3 minutes for "skilled translators", while it is 132 minutes and 75 minutes when with LocStudio. It shows that, EBMTLoc contributes to the translation efficiency to both new hands (55.1% enhancement) and skilled translators (26.3% enhancement). The enhancement rate to skilled translators is different from the one to news hand because skilled translators tend to depend less on TMS but more on their self knowledge.

5 Conclusion

As an attempt at providing a unified statistical model for TMS, the model introduced in this paper enables TMS to provide a translation result under a statistical qualification. As the result, TMS could be expected to gain more success in the translation of the inputs that there are no similar examples in TMEM.

An integrated statistical model is introduced to give a unified statistical qualification in example-based input sentence segmentation, phrase translation selection, best example set selection, and final translation generation.

A prototype TMS EBMTLoc has been developed which adopts a predigested version of the proposed model. As a GTMS radically, EBMTLoc exploits sub-sentential word sequences but not only complete sentences from TMEM to enhance the system coverage, and lows down the matching unit from sentence to continuous word sequences. The translation result could be a complete translation, a partial translation, or no translation.

A series of evaluations have been performed, including absolute evaluation and relevant evaluation for assistant quality and translation efficiency evaluation. As the result, the EBMTLoc system shows obviously better translation assistant quality than traditional TMS, especially to the input sentences when there is no very similar example in TMEM. Compared to employ the traditional TMS, the translation efficiency with the assistance of EBMTLoc could be increased by 26.3%, and this percentage rises to more than 55.1% when the users are unskilled translators.

6 Acknowledgement

We would like to thank Yulin Kang, Ye Zhang and Yue Zhang for their great help on the prototype implementation. In particular, we would like to thank Prof. Chang-Ning Huang and Dr. Jianfeng Gao for their valuable suggestion on this project and the organization of this paper, respectively.

7 References

Christian Boitet. 1994. *(Human-Aided) Machine Translation: A Better Future?* http://cslu.cse.ogi.edu/HLT survey/ch8node5.html#SECTION83

Daniel Marcu. 2001. *Towards a Unified Approach to Memory- and Statistical-Based Machine Translation*. In Proceeding of the Annual Conference of the Association for Computational Linguistics

Emmanuel Planas and Osamu Furuse. 2000. *Multi-level Similar Segment Matching Algorithm for Translation Memories and Example-Based Machine Translation*. In Proceedings of the 18th International Conference on Computational Linguistics

Erik F., Tjong Kim Sang and Sabine Buchholz. 2000. *Introduction to the CoNLL-2000 Shared Task: Chunking*. In the Proceedings of CoNLL-2000 and LLL-2000, p. 127-132, Lisbon, Portugal

Franz Josef Och and Hermann Ney. 2002. *Discriminative Training and Maximum Entropy Models for Statistical Machine Translation*. In Proceeding of the Annual Conference of the Association for Computational Linguistics

Macklovitch, E. and G. Russell. 2000. *What's been Forgotten in Translation Memory*. In Proceedings of the Conference of Association for the Machine Translation in Americas

Ralf D. Brown. 1996. *Example-Based Machine Translation in the Pangloss System*. In Proceedings of the 16th International Conference on Computational Linguistics. p. 169-174

Ralf D. Brown. 1999. *Adding Linguistic Knowledge to a Lexical Example-Based Translation System*. In Proceedings of the Eighth International Conference on Theoretical and Methodological Issues in Machine Translation, p. 22-32

Michel Simard and Philippe Langlais (2001). *Sub-sentential Exploitation of Translation Memories*. Proceedings of Machine Translation Summit VIII. Santiago de Compostela, Spain.

Philippe Langlais and Michel Simard (2002). *Merging Example-Based and Statistical Machine Translation*: An Experiment. Proceedings of the Fifth Conference of Association for Machine Translation in the Americas, Tiburon, California, p. 104-114.

Tony Weale and Andy Way: *Gaijin. 1997. A template-based bootstrapping approach to example-based machine translation*. In Proceedings of "New Methods in Natural Language Processing"

Wei Wang, Ming Zhou, Jin-Xia Huang and Chang-Ning Huang. 2002. *Structure Alignment Using Bilingual Chunking*. In Proceedings of the 19th International Conference on Computational Linguistics

Wang, Wei, Jin-Xia Huang, Ming Zhou and Chang-Ning Huang. 2001. *Finding Target Language Correspondence for Lexicalized EBMT System*. In Proceedings of the 6th Natural Language Processing Pacific Rim Symposium. p. 455-460

Yves LEPAGE. 1998. *Solving Analogies on Words: an Algorithm.* In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL'98) vol. I p. 728-735.