Building Translation Memory System by N-gram

Feiliang REN[1]
Corporate Research Group
Fuji Xerox, Co., Ltd.
Kanagawa, Japan
Feiliang.Ren@fujixerox.co.jp

Shaoming LIU
Corporate Research Group
Fuji Xerox, Co., Ltd.
Kanagawa, Japan
Liu.Shaoming@fujixerox.co.jp

Abstract. In this paper, we proposed a method that built translation memory system by N-gram. It doesn't use any lingual analytical technologies such as chunking, target-text generator, word alignment and so on which are often used in other similar translation memory systems; besides, it has a high proposal speed. These virtues make it is easy to build a translation memory system for those languages that lingual analyses are difficult to progress. We evaluate our system from aspect of assistant efficiency, and experiments show system got a high speed performance; At last, we give some theory compares between our translation memory system and other similar systems.

Keywords: Translation memory, TM, N-gram, translation memory system, translation aid system

1. Introduction

Translation memory(TM) system is a computer aid tool for human translator; it can make human translator's work easy and efficient by reusing existing translated knowledge. It works by comparing the input sentence to be translated with the examples stored in TM database, if succeed, the TM system presents translator with the aligned target sentence.

In earlier TM systems, they provide good translation proposals only when there are some completely matched examples stored in TM database. These systems essentially operate at the level of sentences, and this guarantees a certain level of translation quality, but it can't be used widely because that full-sentence repetition is a rather rare phenomenon.

In response to this problem, TM developers have designed some improved TM systems that operate below the sentence level, or at sub-sentential level.

Brown (1996, 1999) used a full segmentation method to accomplish the sub-sentential TM in his Example-based Machine Translation system. His system first segmented the given input sentence into every possible sequences of words, and then did similarity calculation and search based on these sequences. Different from this full segmentation method, both McTait et al. (1999) and Langlais & Simard (2001) suggested that segmented the input sentence into "linguistically motivated"

[1] Feiliang REN: Assistant of Northeastern University, China, Doctor Candidate. This work was finished during the author worked at the Document Company FujiXerox Co., Ltd. under the Visiting Fellowship Program 8th.

sub-sentential entities were better, at least from a syntactic point of view, than arbitrary sequences of words, and also more likely to lead to useful proposals for the user. And Langlais and Simard (2002) merged the Example-Based system with a statistical engine and used a dynamic programming-based decoder to generate final translation. Jin-Xia Huang et al. (2003) proposed a unified statistical model for TM system, and unified the processes of source string segmentation, best example selection, and translation generation. Similar method also can be found in Marcu (2001) who built a statistical TM to save bilingual word sequences, and produced the translation by using both TM and the statistical model so that the system could exploit the translation knowledge not only at word level but also at phrase level.

All the above improved TM systems general consist of three components: input string's segmentation, sub-sentential search and target-text generator. They are complicated with using much lingual knowledge.

In FujiXerox Co., Ltd., a multi-engines translation aid system is being researched and developed. In this system, there are different translation engines and TM engine that developed by using N-gram technology is the first one of them. This multi-engines system's general chart is shown in Fig.1.

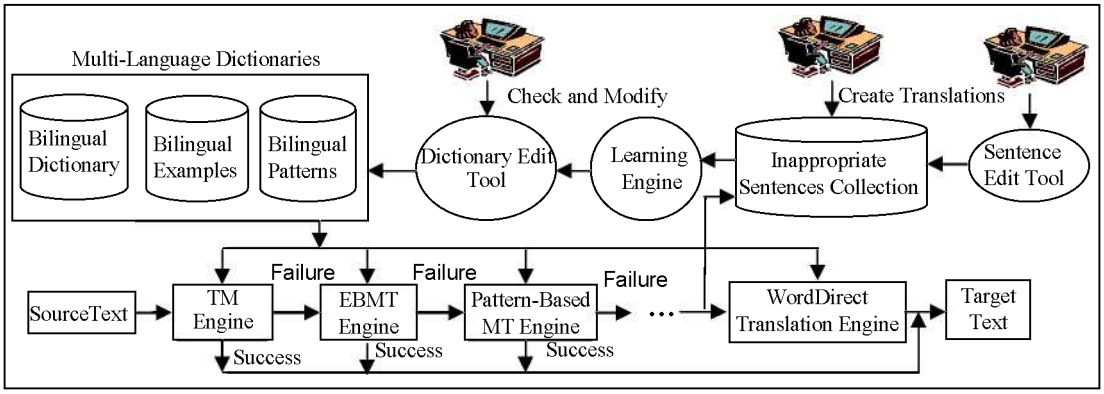

Fig. 1. Multi-Engines Translation Aid System Developed in FujiXerox, Co., Ltd.

The aim of this translation aid system is to provide translation supports for users. And the basic idea for designing this multi-engines translation aid system is: if system can give a confident translation result for the input text, do the translation; otherwise, don't translate it, just present user with hint information! We think a bad and unconfident translation results will confuse the users, and this would be worse than not translate.

In this system, different engines work in serial form. As long as one engine gives an output successfully, the translation procession will be terminated. We design TM engine in this system mainly for the following two tasks:

First, provide translation support for the translation of product demonstrations, software documents, user handbook, idiom, and so on.

Second, provide translation support for the succeeding translation engines. In the succeeding translation engines, system will cut the input text into small segmentations, thus can use TM engine for the translation of these smaller segmentations quickly and exactly.

And TM system designed must satisfy the following conditions: first has a high speed; second, can handle exactly sentence matching; third, can provide similar sentence proposals.

The rest of the paper is organized as following: in section 2, give a detail description of TM system and its functions respectively. System's experiments are shown in section 3. Section 4 is comparison with other TM systems, and finally, in section 5, gives the conclusion of this work.

2. TM System Using N-gram

Given an input string, TM system's procedure will be: 1) has a try on exact translation based on hash function, if success, give the translation result and quit; otherwise, 2) segment the input string into N-gram sequences; 3) provide user some proposal translation results based on the similarity between the input string and examples in translation database, this similarity computation takes N-gram as basic units. System general chart is shown in Fig.2.

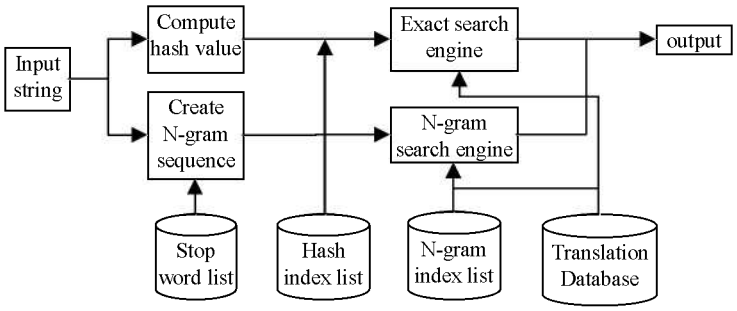

Fig. 2. General Chart of TM Engine System

To describe clearly and conveniently, here first gives some basic definitions used in TM system (in experiments, we built a TM system by using bi-gram (N=2), and other TM systems by using N-gram (N=3, 4, …) have the similar properties with bi-gram system).

- **Definition 1:** We call the set that contains all Chinese and Japanese characters as character set and denoted by S_c.
- **Definition 2**: For character set S_c, any two characters $c_i \in S_c$ and $c_j \in S_c$, we call the arbitrary combination of them $c_i c_j$ or $c_j c_i$ a bi-gram, and denoted by $b_{i,j}$ or $b_{j,i}$.
- **Definition 3**: For a given sentence $S = (c_1 c_2 c_3 ... c_i ... c_n), \forall c_i \in S_c$, we call the successive, adjacent bi-gram sequence $(b_{1,2} b_{2,3}, ..., b_{i,i+1}, ... b_{n-1,n})$ as S"s bi-gram sequence, and denoted by $Seq(S)$.

From these definitions, we can know: 1) for an input string, we consider only Chinese and Japanese characters, and all other symbols will be omitted. 2) A bi-gram is an arbitrary combination of two Chinese or Japanese characters. 3) Bi-gram sequence for an input string is consecutive.

2.1 Sentence Level Exact Matching

Provide user translations based on sentence-level exact matching is the basic function for a TM system. Given an input string, system first compute its hash value according a hash function, then search the translation database with the help of hash index list, if search successfully, provide user the corresponding translation result; otherwise, go to next step described in section 2.2.

2.2 Sub-sentential Level Proposal

Sub-sentential level proposal is based on bi-grams. It consists of the following three steps:

1. Segment input string into bi-gram sequence;
2. Search translation database to find some translation proposal candidates based on the bi-gram entities in bi-gram sequence;
3. Select from proposal candidates to find some final translation proposals and present them to user;

Figure 3 is an example of the bi-gram segmentation.

Input string: 他们为年轻付出的代价很大
Bi-gram sequence: <他们> <们为> <为年> <年轻> <轻付> <付出> <出的> <的代> <代价> <价很> <很大>

Fig. 3. An Example of bi-gram segmentation

After system gets the segmentation sequence for the input string, next step is to search similar translation examples in translation database based on this segmentation sequence. System will retrieve all translation pairs whose source language part contain at least one bi-gram of the input string and take these translation pairs as candidates for next step. This is to say, every translation pair SP_i will be selected as candidates if its source language part $SP_i(Source)$ satisfies the flowing condition: $\exists b_{i,j}, b_{i,j} \in \{Seq(Input) \cap Seq(SP_i(Source))\}$.

After the retrieval process, system will rank these selected sentence pairs and select some best matching pairs and return their corresponding translations as input sentence's proposal results. There are many measures for ranking the selected sentence pairs: such as the *edit distance* and *longest common substring* and so on. To make full use of bi-gram structure, here system uses bi-gram coverage metric as rank function, and the formula is shown as following:

$$Coverage\ (S_1, S_2) = \frac{2 \times SameBigram\ (S_1, S_2)}{BigramLen\ (S_1) + BigramLen\ (S_2)} \quad (1)$$

$BigramLen(S)$ is bi-gram number in sentence S ; $SameBigram(S_1, S_2)$ is the number of bi-grams that appear both in sentence S_1 and in sentence S_2 .

2.3 Sentence Pattern Proposal

From users' viewpoint, sometimes they may not want to a whole sentence's proposals, but just want to make sure some special usages of some words. For example, they may just want to know how to

translate the word w_1 when it appeared before word w_2, and separated by some other words at the same time. Therefore we think it will be very useful if the system can provide some pattern proposal for user.

In our TM system, some following-alike sentence patterns are also stored in translation database:

Example: V1-ConsecutiveForm＋ながら…V2…

一边V1…一边V2…

By extending the translation database from containing sentence pairs only to containing both sentence pairs and sentence pattern pairs, we improve the system's proposal ability.

3. Experiments

We implemented a Chinese-Japanese TM system to evaluate the performance of our approach. System interface is shown in Fig.4. And some basic statistical information about the experiment corpus is shown in table 1. The computer used is P_{IV} 3.2GHz , 1G Memory.

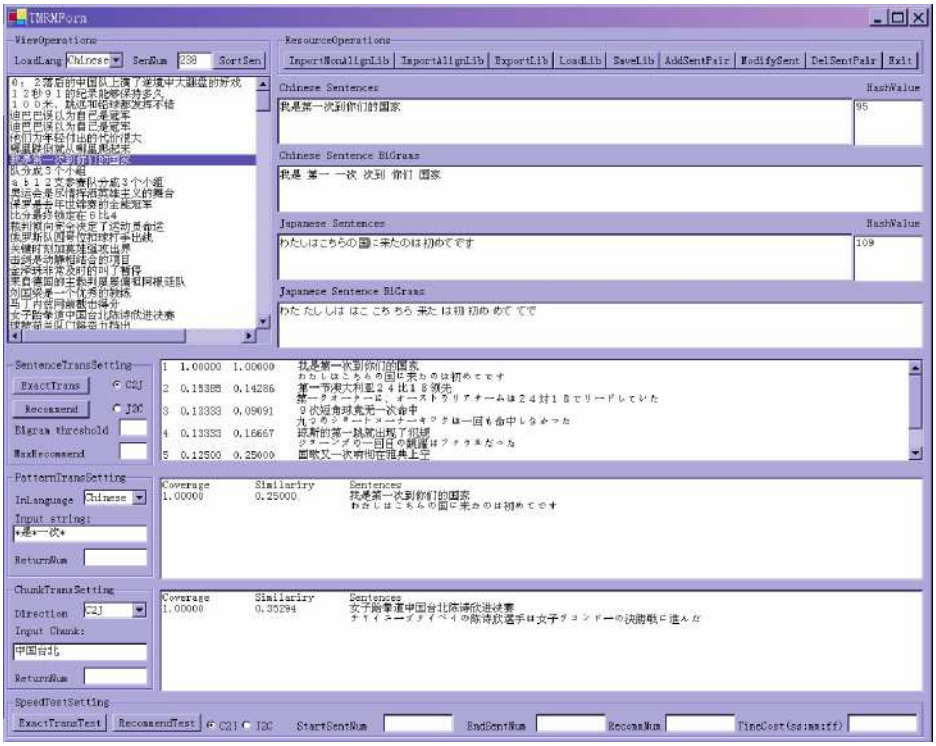

Fig. 4. The Interface of Developed TM System

Table 1. Basic Statistical Information about Translation Database

Total Number of Chinese-Japanese Sentence Pairs	10080
Total Number of Chinese Characters in Corpus	174895
Total Number of Japanese Characters in Corpus	283845
Average Characters Length of a Chinese Sentence	17.35
Average Characters Length of a Japanese Sentence	28.16

Our experiments test the system's performance mainly from time aspect, and it consist of two parts: first part is to test efficiency on extract matching, and the other part is on sub-sentential proposal. We compare the N-gram method with a tree-search method which developed by FujiXerox Co., Ltd. and a word-index method which use word as index and search unit. Experiments results are shown in table2.

Table 2. Experiments about time performatnce

Method	Experiment Data	Space Used	Has extract translation function?	Speed (ms/sent)	Has proposal function?	Speed (ms/sent)
Word-index method	150,000 sentence pairs	750MB	○	80	○	80
Tree-search method	300,000 sentence pairs	298MB	○	0.047	×	
N-gram method(N=2)	10,000 sentence pairs	2.16M	○	0.0015	○	0.702

(Note: "○" denotes having this function, "×" denotes not having this function)

Form table2 we can see, use word index method, system can complete the tasks of both translation and proposal, but the speed is lower compared with N-gram method. Tree search method gets a high speed performance, but it can't complete the task of proposal. Our N-gram method can complete both the translation task and proposal task with a higher speed compared with the other two methods.

4. Compare with Other TM Systems

Because of the different evaluation methods and different evaluation criterions for a TM system, we can't compare our TM system's performance with other TM systems directly. But we still can do some comparisons in theory.

For a given string which has n characters, using Brown's full segmentation method, it will create $\frac{n(n+1)}{2}$ segmentations; in our method, it only will create $n-1$ segmentations at most. Thus, if use same search method, Brown's method will use more time than ours.

For some other segmentation methods as used by McTait et al. and Langlais & Simard who segmented the input string into linguistically motivated sequences, they will get the least segmentations, but they had to use a linguistically motivated segmentation method, which will slow down the whole system greatly. For those TM systems that used statistical method, because of the complicated computations needed in them, the time cost won't be low. This can be proved from the experiment results in Jin-Xia Huang et al. (2003).

Form above analyses, we can see that our TM system has a speed superiority compared with other similar TM systems. Besides the high speed, in our TM system, we don't use any lingual analytical technologies. This simply the developing process of a TM system greatly, especially for the languages that these lingual analytical technologies are difficult.

At the same time, by extending the translation database from sentence pairs to sentence pairs and sentence pattern pairs, our TM system also can provide user some sentence pattern proposals which we think will be very useful for dealing with complicated sentences.

5. Conclusions

In this paper, we proposed a TM system that using N-gram technology. It has the virtues of easily being built and high proposal speed. It accomplishes the task of exact matching and sub-sentential proposals based mainly on N-grams. It doesn't need any lingual analytical technologies such as chunking, target-text generator, and word alignment and so on, this is very important for some languages that these lingual analyses are difficult to progress.

References

[1]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

[2]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

[3]Kevin McTait, Maeve Olohan, and Arturo Trujillo. 1999. A Building Blocks Approach to Translation Memory. In Proceedings of the 21st ASLIB International Conference on Translating and the Computer,London, UK.

[4]Emmanuel Planas. 2000. Extending Translation Memories. In EAMT Machine Translation Workshop, Ljubljana, Slovenia, May.

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

[6]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.

[7]Jin-Xia Huang, Wei Wang and Ming Zhou (2003). A Unified Statistical Model for Generalized Translation Memory System. MT Summit IX; http://www.amtaweb.org/summit/MTSummit/papers.html

[8]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

[9]Michel Simard, 2003, Translation Spotting for Translation Memories. HIT-NAACL 2003 Workshop: Building and Using Parallel Texts Data Driven Machine Translation and Beyond, pp.65-72. Edmonton, May-June, 2003