The TRGTK's System Description of the PatentMT Task at the NTCIR-10 Workshop

Hao Xiong*
Torangetek Inc./Key Lab. of Intelligent Information Processing
Institute of Computing Technology, CAS
P.O. Box 2704, Beijing 100190, China
xionghao@torangetek.com

Weihua Luo
Torangetek Inc./Key Lab. of Intelligent Information Processing
Institute of Computing Technology, CAS
P.O. Box 2704, Beijing 100190, China
luoweihua@torangetek.com

ABSTRACT

This paper introduces the TRGTK's system for Patent Machine Translation at the NTCIR-10 Workshop. In this year's program, we participate Chinese-English, English-Japanese and Japanese-English three subtasks. We submit required system results for Intrinsic Evaluation (IE), Patent Examination Evaluation (PEE), Chronological Evaluation (ChE), and Multilingual Evaluation (ME). Different from last year's strategy, we focus on developing a strong and practical system for large-scale machine translation requirements. We design parallel algorithm for Chinese word segmentation, weights tuning and translation decoding, especially we propose a documental level translation method to improve the translation quality of special terms. Experimental results show that our system reduce the training and decoding time while still achieve promising translation results.

Categories and Subject Descriptors

H.4 []: Artificial Intelligence; D.2.8 [**Natural Language Processing**]: Machine Learning—*Machine Translation, Patent Translation, Parallel Computing*

General Terms

Experiment

Keywords

Machine Translation, Patent Translation, Parallel Computing

Team Name

TRGTK(Torangetek Inc.)

Subtasks

Chinese-English
Japanese-English
English-Japanese

*We are a start-up company incubated by Institute of Computing Technology and aim to supply high quality machine translation service.

1. INTRODUCTION

This year's Patent Machine Translation task[4] at the NTCIR-10 workshop consists of C-E, J-E and E-J three subtasks. Different from last year's program, the organizers distribute several testing corpus for four types of evaluations: Intrinsic Evaluation (IE), Patent Examination Evaluation (PEE), Chronological Evaluation (ChE), and Multilingual Evaluation (ME). We participate all kinds of subtasks and submit all required system results.

Since we are a commercial company, our goal is to supply fast and reasonable machine translation service for Translation Aided Company. Thus in this year, we concentrate on developing a strong and practical system for large-scale machine translation requirements which is quite different from other participants where they mainly focus on improving the translation quality. According to previous work, hierarchical phrased based(HPB) translation model [1] is well studied and achieves promising results in previous machine translation evaluations. Most importantly, HPB is easy to implement and extend. However, since HPB use CKY algorithm to search the best translation, its decoding time is unsatisfied when long translation sentence given. To reduce the time consuming while decoding, we record searching stacks for some high frequently translated phrases, and search the stacks in parallel. Moreover, we cluster related sentences into similar documents and design a documental CKY algorithm to translate them simultaneously in order to reduce the searching time for similar phrases and obtain consistent translation for some special terms such as patent terms. Furthermore, in the pipeline of machine translation, preprocessing step as Chinese word segmentation is necessary but time consuming, therefore, we also use parallel algorithm to accelerate its decoding via computing on GPU because of its numerous computing units and cheap price. Also, we reduce training time by performing rule extraction and word alignment in parallel. Large scale experiments show that when using parallel computing on 100 CPU, training time includes word alignment and weights tuning for 10 million bilingual corpus is reduced to 8 hours and translation decoding speed is 1000 words per second which is qualified to large scale translation requirements.

The remainder of this paper is organized as follows, we illustrate overall system architecture along with detailed technical descriptions in Section 2. Section 3 mainly presents our

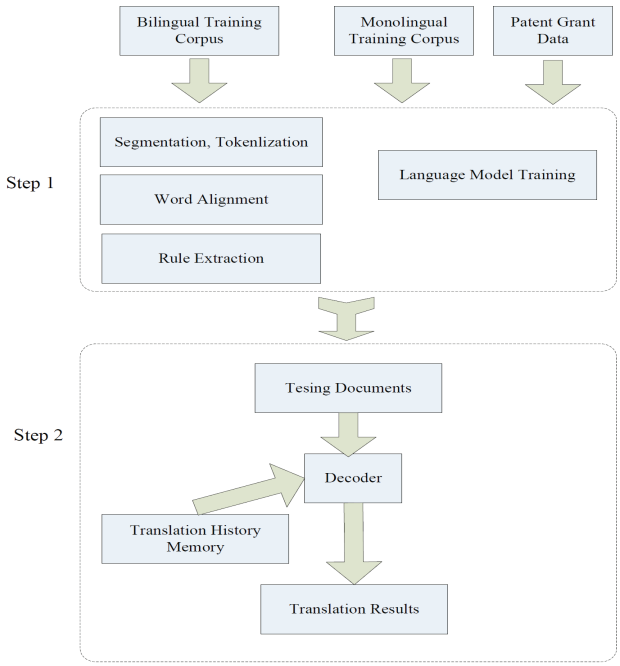

Figure 1: The overall architecture of our system, where step 1 is the corpus preprocessing stage preparing for the step 2, in which testing sentences are clustered into several documents and translated using document-level decoding algorithm.

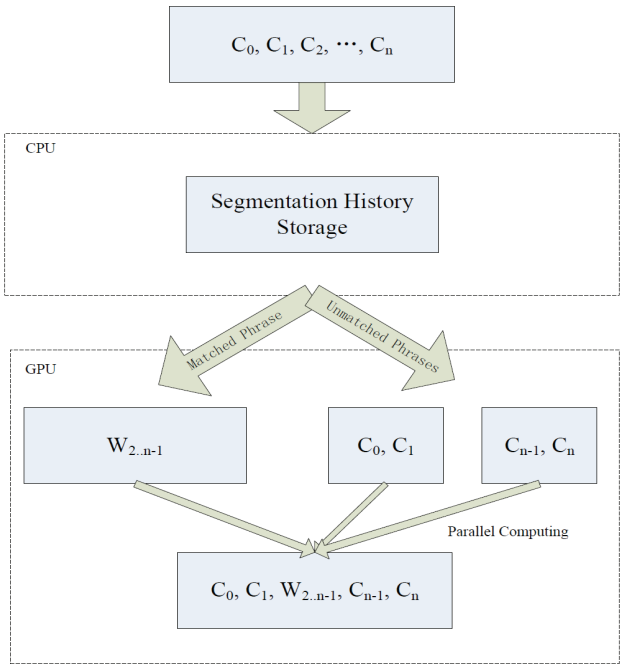

Figure 2: The detailed process of our segmenter, where we first search the segmentation schema based on the matching of historical segmentation storage, and then parallel search the remaining phrases whose segmentation are unknown after the first matching process.

experimental results, and we conclude our paper in Section 4.

2. SYSTEM ARCHITECTURE

We illustrate system architecture in figure 1, where the whole process includes two steps including corpus preprocessing and translation generating, respectively. Most of our techniques in this architecture are similar to the Moses toolkit's [1], thus in this section, we will mainly focus on describing the special designed parts of our system while temporarily omit other skills which will be later introduced in the experimental section.

2.1 Chinese Word Segmentation on GPU

For natural language applications, word segmentation is a fundamental research and is necessary for some advanced applications such as Information Searching, Question Answering and Machine Translation. Traditional research on word segmentation method treat it as the problem of sequence labeling and could be solved via three steps: feature extraction, class prediction and results searching. In these three steps, the previous two steps could be computed offline and hence does not consume time in practical applications, thus we concentrate on developing parallel algorithm for the step of results searching. Generally, most researchers utilize dynamic algorithm to find the best segmentation schema where the global optimal solver based on the local optimal solution. Under the instruction of dynamic algorithm, we could first divide sentence into multiple phrases segmented in parallel locally, and then combine them to obtain global segmentation schema recursively. On the other hand, in recent years, following the tendency of Big Data[2], computing on GPU[6] become increasingly popular benefits from its numerous computing units and cheap price. Here, we also perform our segmenter on GPU in order to obtain extremely computing speed.

Concretely, figure 2 gives a detailed process of our parallel segmenter. Our system consists of two parts where the first part is computed on the CPU and used for searching the segmentation schema based on the historical segmentation of the training sentences in advance. This part is very useful in large scale application in that it can largely reduce the segmentation time for some common sentences. For instance, assuming we want to segment the sentence consists of characters "*abcdefghijk*", and the history storage has the segmentation schema "*de fg hi*". In this case, we should just search the segmentation for phrases "*abcde*" and "*hijk*" since the segmentation of phrase "*fg*" is known when given the context "*de*" and "*hi*". It is worth noting that the traditional approaches for segmentation consider maximal two characters before and after the current character, thus the segmentation of phrase "*de*" in this example is unknown and should be further searched with the left context "*bc*". Another part of our system is designed for parallel segmentation of the remaining phrases whose segmentation are unknown according to previous matching process. In this part, we first divide the remaining phrases into several units which

[1] `http://www.statmt.org/moses_steps.html`

[2] `http://en.wikipedia.org/wiki/Big_data`

2 words	3 words	4 words	5 words	6 words
n+n	n+n+n	n+n+n+n	v+v+n+n+n	n+n+c+v|n+n+n
n+v	v+n+n	n+n+v+n	d+v+n+n+n	n+n+v|n+c+v|n+n
v+n	n+v+n	v+n+n+n	m+v+m+n+n	n+n+u+b+v|n+n
a+n	v+v+n	v+n+v+n	b+v+n+v+n	v|n+n+v|n+c+v|n+n
d+n	b+v+n	n+v+v+n	n+n+v+n+n	n+v|n+u+n+v|n+n
b+n	n+m+n	v+v+n+n v+n+b+n	a+n+v+n+n	

Table 1: Rules used for generating candidate string for term extraction where a is adjective, b is distinguish word, c is conjunction, d is adverb, n is noun, m is numbers, v is verb, u is particle, v|n is verb or noun.

just contain five characters and transmit them to the computing units in GPU. Since the size of context window for features extraction is maximal five, we could parallel predict the label of each unit independently. After generating each unit, we then hierarchically search the best solution in parallel. Since most of words in Chinese generally consists of not more than five characters, we combine sequential five units into one big unit and then search the optimal solution of this unit. Afterwards, we combine two units to search the segmentation of more characters. This process is recursive and could be easily computed in parallel. Later in the experimental section, we will show that after computing in parallel, the time used for segmentation is largely reduced. 1.

2.2 Document-level Decoding

In practical, large number of translation requirements such as patent translation, stem from documental translation. Documental translation is characterized by the followings: firstly special terms like patent terms in the same document should have consistent translation, second is that some phrases will appear repeatedly in different sentences, and lastly contextual knowledge for similar phrases in different sentence potentially help the selection the better rules. Thus it is reasonable to develop a documental decoding algorithm that fast and effectively translate overall sentences using present parallel computing technology. However, to design a documental level translation approach, there are several problems need to be addressed: the first is how to cluster related sentences into one document, second is how to recognize phrases like patent terms that should have the consistent translations, last is how to modify original CKY decoding algorithm to fast generate translation among huge number of searching stacks.

Document generation. Although the provided patent documents contain document IDs which could be used to recognize sentences in one document, for the sake of generality, we use KNN algorithm[2] to cluster sentences into one document according to their lexical similarity. To measure the similarity between two sentences, we follow the traditional bag-of-words approach that has been applied in related tasks[5] which we consider the 5,000 most frequent words $w_1, ..., w_{5000} \in W$ as dimensions of a vector space and define sentences as vectors using frequency of frequent words:

$$\overrightarrow{S} = (freq(w_1), ..., freq(w_{5000})) \quad (1)$$

Given the vector representations of two sentences, we calculate their similarity as the cosine of the angle between the two vectors:

$$sim(S_1, S_2) = \frac{\overrightarrow{S_1} \cdot \overrightarrow{S_2}}{|\overrightarrow{S_1}| * |\overrightarrow{S_2}|} \quad (2)$$

In our experiments, we control the size of sentences in each document lower than 50 in order to guarantee sentences in one document are highly related.

Term Recognition. To measure one candidate string is a term or not, we follow the traditional approach that use *C-value* to measure the termhood, it defines as following

$$\textit{C-value}(a) = \begin{cases} log_2|a| \cdot f(a), & a \textit{ is not nested}, \quad (3a) \\ log_2|a|(f(a) - \frac{1}{P(T_a)} \sum_{b \in T_a} f(b)) & (3b) \end{cases}$$

where a is the candidate string, $f(a)$ is its frequency of occurrence in the corpus, T_a is the set of extracted candidate terms that contain a, $P(T_a)$ is the number of these candidate terms. Readers can refer to [3] for detailed explanation. And the rules used for generating candidate string is shown in table 1.

Documental CKY Algorithm. In our decoding stage, different from traditional CKY algorithm where each word utilize independent searching stack, we push all similar patent terms or similar phrases with length more ten words into one searching stack. For better understanding, we illustrate the decoding process in figure 3. The primary modification in our methods is that when computing a phrase involves associated patent terms, the selection of its translation is not only depending on adjacent words but the context of its associated terms in another sentences will also be considered. Thus, we can assure the similar patent terms select the same translation, and the selected translation has considered different context in each sentences.

2.3 Translation History Memory

The function of this part is the same as segmentation history storage in our segmenter which we first translate huge number of sentences and then record searching stacks for high frequent translated phrases. When new testing sentence is given, we could reduce the searching time in CKY algorithm conditioned on the number of matched translated phrases. In our experiments, we run training set to generate the historical translation.

3. EXPERIMENTS

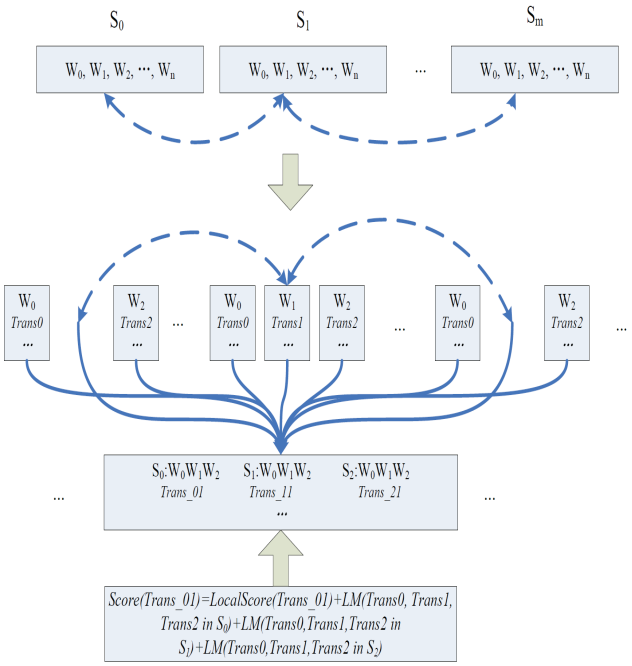

Figure 3: The decoding process for document, where S_0, S_1 and S_m are sentences in document, and w_0, w_1, w_n are corresponding words in these sentences. Dashed lines connect the similar recognized patent terms. *LocalScore* is computed by local features such as lexical probability, rule size etc,. *LM* is calculated by language model given its context.

3.1 Data Usage

The organizers provide both bilingual patent description sentence pairs for each subtasks as well as monolingual patent grant documents for Japanese and English[4]. However, provided monolingual corpus contain large number of sentences but some is ancient to testing set, thus we just take some portion of them to train language model. The overall corpus we use in our system is presented in table 2.

System	Bilingual	Monolingual
C-E	1 Million	40 Million
J-E	3 Million	40 Million
E-J	3 Million	73 Million

Table 2: The overall corpus we used in our system, wherein monolingual corpus are used to train language model.

3.2 Preprocessing of Japanese

Japanese is a kind of agglutinative languages. Its biggest characteristic is to indicate the grammatical relations in a sentence by means of adhering function words to behind of notional words. There are no obvious boundaries between words in Japanese. So, Japanese word segmentation is one of necessary procedures on machine translation of Japanese-to-English. Preprocessing on Japanese corpus in out task consists of two procedures.

1. Full-width characters converting to half-width ones: In computer editorial process, letters, numbers and symbols may appear in half-width or full-with forms. This phenomenon will affect phrases' identification in the translation process in some extent, which may reduce the translation quality in the end. So we converted full-width characters to half-width ones in corpus in the first step.

2. Japanese word segmentation: Japanese word segmentation is basic task of Japanese information processing, which is also the foundation of Japanese machine translation. We used Chasen (chasen-2.4.4)[3], one of the most famous open source Japanese lexical analysis tools, to do the task of Japanese word segmentation in the second step. Chasen is developed by Nara Institute of Science and Technology, which is based on Hidden markov model.

3.3 Results on Developing Set

We use the given developing corpus as our developing set, and use the SRI Language Modeling Toolkit [7] to train the Japenese/English 5-gram language model with Kneser-Ney smoothing on the Japenese/English side of the training corpus in addition with corresponding monolingual corpus. Noting that, we do not use English sentences in J-E task to train language model for C-E translation and vice versa. We use SyMGiza[4]to generate the word alignment. Table 3 gives the experimental results on developing set.

C-E	J-E	E-J
33.59	27.08	33.89

Table 3: Experimental results on developing set.

3.4 Large Scale Experiments

Parallel Segmenter. To test the speed of our parallel segmenter, we perform several compared experiments on C-E translation, first type, namely ***sys1***, use segmentation historical storage and ***sys2*** does not. Table 4 lists the time consuming for segmenting 1 million training data. Noting that we use additional 10 million Chinese sentence to generate the historical segmentation storage. It is clear that when more computing units given, the time consuming decrease significantly. Another funding is that historical storage also help to reduce the time consuming benefits from that our additional corpus is also from patent domain.

System	GPU	Time(minutes)
sys1	500	0.3
sys1	1	32
sys2	500	0.5
sys2	1	40

Table 4: Experimental results for segmenter.

Parallel Translation In this experiment, we evaluate the whole time for training 10 million bilingual corpus which

[3]http://chasen-legacy.sourceforge.jp/
[4]http://psi.amu.edu.pl/en/index.php?title=SyMGIZA&redirect=no

includes segmentation, tokenlization, word alignment, rule extraction, and weights tuning. We also evaluate the time for decoding the testing set using translation memory from training set. Table 5 shows the detailed results given different configurations for parallel translation. We can find when given 100 CPU for training, the time drops to 8 hours which is convenient for large number of experimental attempting. And the speed of parallel decoding is very fast even with no historical translation memory which is applicable for large scale translation requirements.

System	CPU	Time
parallel training	100	8 hours
training	1	45hours
parallel decoding	100	800words/sec
parallel decoding(Memory)	100	1000words/sec
decoding	1	50words/sec

Table 5: Experimental results for parallel translation, where storage means using historical translation records.

3.5 Final Results

Table 6 present our system results on final testing set.

System	C-E	J-E	E-J
IE	**34.63**	**26.99**	**32.21**
ChE	33.46	26.34	31.4
ME	21.52	26.34	

Table 6: Evaluation results of our final submission.

4. CONCLUSION

In this paper, we summarize techniques we used in this year's evaluation tasks. We participate three subtasks of Patent Machine Translation task and submit six systems for each subtask. Our goal is to develop a fast and applicable translation system for large scale patent translation requirements. Experimental results show that when using parallel computing, our system could deal with 10 million corpus in 8 hours, which do help researchers for fast experimental attempting.

5. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

6. REFERENCES

[1] D. Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics, 2005.

[2] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[3] K. Frantzi, S. Ananiadou, and J. Tsujii. The c-value/nc-value method of automatic recognition for multi-word terms. *Research and Advanced Technology for Digital Libraries*, pages 520–520, 1998.

[4] I. Goto, K. P. Chow, B. Lu, E. Sumita, and B. K. Tsou. Overview of the patent machine translation task at the ntcir-10 workshop. In *Proceedings of the NTCIR-10 Workshop*, 2012.

[5] W. Guo and M. Diab. Semantic topic models: Combining word distributional statistics and dictionary definitions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 552–561. Association for Computational Linguistics, 2011.

[6] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips. Gpu computing. *Proceedings of the IEEE*, 96(5):879–899, 2008.

[7] A. Stolcke. Srilm - an extensible language modeling toolkit. In *Proceedings of ICSLP*, volume 30, pages 901–904, 2002.