Example-Based Machine Translation Based on Tree-String Correspondence and Statistical Generation

ZHANYI LIU, HAIFENG WANG and HUA WU
Toshiba (China) Research and Development Center
501, Tower W2, Oriental Plaza, No.1, East Chang An Ave., Dong Cheng District
Beijing, 100738, China
{*liuzhanyi,wanghaifeng, wuhua*}*@rdc.toshiba.com.cn*

Abstract. This paper describes an example-based machine translation (EBMT) method based on tree-string correspondence (TSC) and statistical generation. In this method, the translation example is represented as a TSC, which is a triple consisting of a parse tree in the source language, a string in the target language, and the correspondence between the leaf node of the source-language tree and the substring of the target-language string. For an input sentence to be translated, it is first parsed into a tree. Then the TSC forest which best matches the input tree is searched for. Finally the translation is generated using a statistical generation model to combine the target-language strings of the TSCs. The generation model consists of three features: the semantic similarity between the tree in the TSC and the input tree, the translation probability of translating the source word into the target word, and the language-model probability for the target-language string. Based on the above method, we build an English-to-Chinese MT system. Experimental results indicate that the performance of our system is comparable with phrase-based statistical MT systems.

Keywords: example-based machine translation, translation example, tree-string correspondence, statistical generation

1. Introduction

This paper presents an Example-based machine translation (EBMT) method based on tree-string correspondence (TSC) and statistical generation. According to this method, the translation example is represented as a TSC, which is a triple consisting of a parse tree in the source language, a string in the target language, and the correspondence between the leaf node of the source-language tree and the substring of the target-language string. There are three steps to translate an input sentence. First, the input sentence is parsed into a tree. Then the TSC forest that best matches the input tree is searched for. Lastly, the translation is generated using a statistical generation model to combine the target-language strings in the TSCs. The generation model consists of three features: the semantic similarity between the tree in the TSC and the input tree, the translation probability of translating the source word into the target word, and the language-model probability for the target-language string.

Based on the above method, we built an English-to-Chinese MT system. Experimental results indicate that the performance of our system is comparable with phrase-based statistical MT systems.

The paper is organized as follows. In the remainder of this section, we summarize the related work and present an overview of our EBMT method based on TSC and statistical generation. In Sect. 2, we give a detailed description of TSC. In Sect. 3, we describe the tree matching method. In Sect. 4, we present the statistical translation generation method. In Sect. 5, we describe the resources used in the experimental set up. Section 6 presents the results of these experiments together with a description of the evaluation carried out. In Sect. 7, we present our conclusions together with avenues for further research.

1.1. Translation example storage

In recent years, EBMT systems have shown an increasing tendency to use annotated tree structures as translation examples (Watanabe 1992; Poutsma 2000; Al-Adhaileh et al. 2002; Way 2003; Aramaki and Kurohashi 2004). In the preprocessing stage of these systems, both the source sentence and the

target sentence in an example are parsed into trees. Moreover, the corresponding subsentential nodes and word pairs in the examples are aligned. During translation, bilingual fragments are extracted from the preprocessed examples and are used to generate the final translation. As the annotated example contains additional syntactic information, such as phrase-structure information, these annotated translation examples can be used to improve translation accuracy. However, it is difficult to build such a structural translation example database because of two reasons. First, some languages lack parsing tools of high accuracy. Second, it is difficult to build the structural correspondences between two languages with quite different linguistic structures (Shieber 1994; Al-Adhaileh et al. 2002), such as English and Chinese.

In this paper, the translation example is represented as a TSC. To convert a bilingual sentence pair into a TSC, only the source-language sentence should be parsed. The target-language sentence is represented as a word string (cf. Yamada and Knight 2002; Langlais and Gotti 2006). Moreover, only the word-level correspondences are stored in the TSC. The word correspondences can be obtained from bilingual word alignment.

1.2. Translation example matching

In an EBMT system using structural translation examples, the similarity between the translation example and the input sentence is calculated using tree-edit distance (Matsumoto et al. 1993; Watanabe 1995; Al-Adhaileh and Kong 1999). The calculation involves a rather complex tree-matching operation. It is certainly a considerable computation cost in any practical system (Somers 1999). Utsuro et al. (1994) attempted to reduce the computational cost of matching by taking advantage of the surface structure of Japanese, in particular its case-frame-like structure (NPs with overt case marking). They developed a similarity measure based on a thesaurus for the head nouns. Their method unfortunately relies on the verbs matching exactly, and also seems limited to Japanese or languages with similar structures.

In this work, we propose a greedy TSC-tree matching algorithm to find the TSC forest efficiently. In this algorithm, we need to search out only the TSC forest that matches the input parse tree to translate an input sentence. The corresponding translation fragment in the translation example is also determined by the TSC forest.

1.3. Translation generation

For EBMT systems, there are two major approaches to selecting the translation fragments and to generating the final translation. Semantics-based approaches (Aramaki et al. 2003; Aramaki and Kurohashi 2004) obtain an appropriate translation fragment for each part of the input sentence by means of a thesaurus. The final translation is generated by combining the translation fragments in a predefined order. This approach does not take into account the transition between fragments. Statistical approaches (Kaki et al. 1999; Callison-Burch and Flournoy 2001; Akiba et al. 2002; Imamura et al. 2004) select translation fragments with a statistical model. The statistical model can solve the transition problem by using n-gram co-occurrence statistics. However, this method does not take into account the semantic relations between the example and the input sentence.

We propose a method that combines the semantics-based and the statistical approaches to select the translation fragments and to generate the translation. The generation model consists of three feature functions: the semantic similarity between the tree in the TSC and the input tree, the translation probability of translating the source word as the target word, and the language-model probability for the target language string.

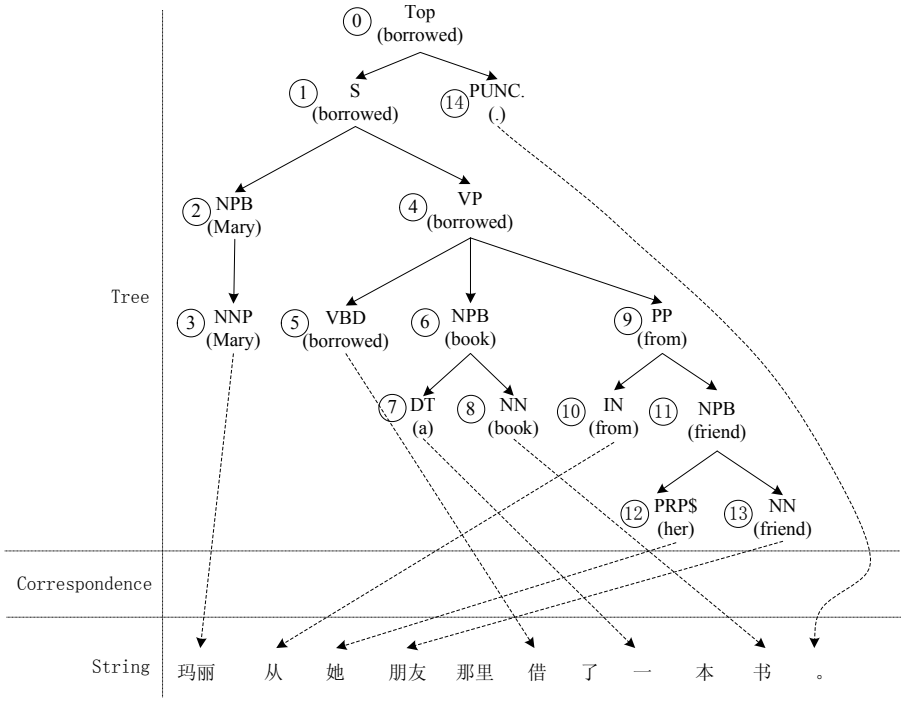


Figure 1. Example of a TSC for example (1)

2. Tree-string correspondence

2.1. Matching-tree

Given a phrase-structure tree T and a sub-tree T_s of T, T_s is a *matching-tree* of T if T_s satisfies the following conditions:

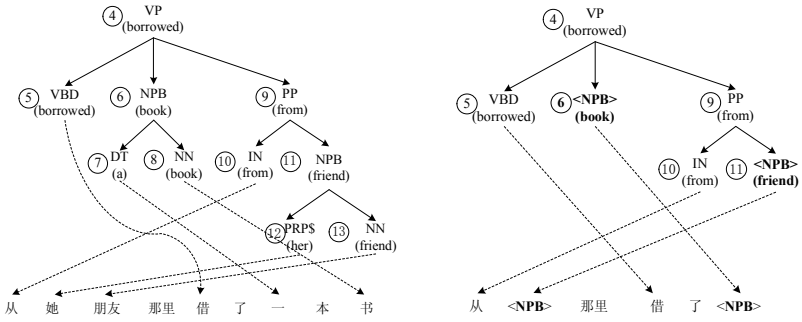
1. There is more than one node in T_s;
2. In T_s, there is only one node r (the root node of T_s) whose parent node is not in T_s. All the other nodes in T_s are descendant nodes of r;
3. For any node n in T_s except r, the sibling node of n is also in T_s.

If all the descendant nodes of root node r are in T_s, then the matching-tree T_s is a *maximal matching-tree* rooted at r. Otherwise the T_s is a *partial matching-tree* rooted at r. A tree T is a maximal matching-tree of itself.

In this article, each node of a tree is labeled with its headword and category.

Figure 1 shows a TSC corresponding to the translation example (1). The concept of TSC will be presented in detail in Sect. 2.2. Here, let us consider the tree T_0 rooted at node 0 in Fig. 1 (for the sake of clarity, each node of the tree is numbered in ascending order). The subtree that consists of nodes 4-13 is a maximal matching-tree of T_0. The subtree that consists of nodes 4-6 and 9-11 is a partial matching-tree of T_0. However, the subtree that consists of the nodes 4-6 is not a matching-tree of of T_0, because the sibling node 9 of nodes 5 and 6 is not in the subtree.

(1) Mary borrowed a book from her friend.
玛丽从她朋友那里借了一本书。
Mali cong ta pengyou nali jie le yi ben shu .
MARY FROM HER FRIEND THERE BORROW PAST ONE COUNT BOOK



a. Maximal TSC b. Partial TSC

Figure 2. Maximal TSC and partial TSC

2.2. Definition of tsc

A TSC is defined as a triple $< t, s, c >$, where t is a matching-tree of the source-language parse tree; s is a target-language string corresponding to t; c denotes the word correspondence, which consists of the links between the leaf nodes of t and the substrings of s.

A TSC is used to represent either of the following:

1. A static translation example. In the TSC-based EBMT system, a preprocessed translation example is statically stored as a TSC in the example database;

2. A dynamic translation example fragment. During the example matching procedure, a translation example fragment, which is identified to match the input, is represented as a TSC.

In this article, we use English-ChineseMT as a case study, and the English-Chinese translation in (1), the TSC of which appears in Fig. 1, as a working example.

In Fig. 1, the English sentence is parsed into a phrase-structure tree rooted at the node 0. Each node of the tree is labeled with its headword and category. For example, the root node 0 of the tree is labeled "TOP (borrowed)", where "Top" is the category and *borrowed* is the headword. The Chinese sentence is segmented into words. The corresponding words between the English sentence and the Chinese sentence are aligned, as represented by the broken lines in the figure. For some words in one language, such as the Chinese word 那里*nali* 'there' and 了*le* (an aspect marker), there are no corresponding words in the other language.

According to the definitions of maximal matching-tree and partial matching-tree, we also define two kinds of TSC: *maximal TSC* and *partial TSC*.

2.3. Maximal tsc

Given a TSC $< t_0, s_0, c_0 >$, TSC $< t, s, c >$ is a *maximal TSC* of $< t_0, s_0, c_0 >$ if t is a maximal matching-tree of t_0. The target language string s is determined as follows:

Let $s_0 = e_1, e_2, \ldots, e_I$. S_t is the word set that includes all the words in the target language corresponding to the leaf nodes in t. In the set S_t, we regard the same words at different positions as different words.

Let $L = \text{argmin}_j\{j|e_j \in S_t\}$ and $R = \text{argmax}_j\{j|e_j \in S_t\}$.

Then, for the maximal TSC $< t, s, c >$, $s = e_L, e_{L+1}, \ldots, e_R$. In this article, s is also regarded as the translation of the root node of t.

In Fig. 2a, e_L is 从*cong* 'from', e_R is 书*shu* 'book'. s is the string (2), and is also regarded as the translation of node 4.

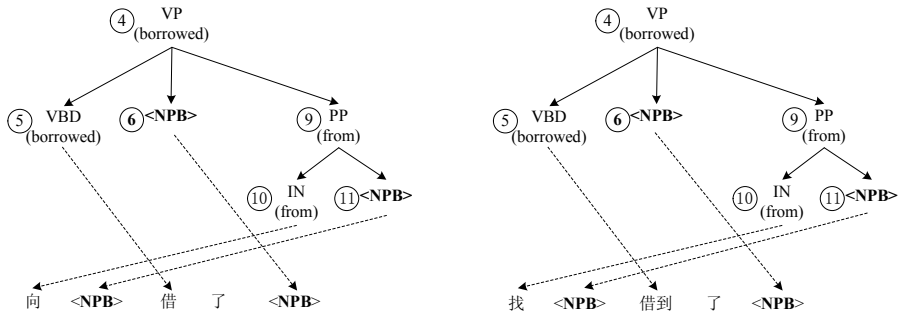


Figure 3. Example of homologous TSCs

(2) 从她朋友那里借了一本书
cong ta pengyou nali jie le yi ben shu
'borrow a book from her friend'

In the maximal TSC $< t, s, c >$, c represents the correspondence between the leaf nodes of t and the words in s.

2.4. Partial tsc

Given a TSC $< t_0, s_0, c_0 >$, TSC $< t, s, c >$ is a *partial TSC* of $< t_0, s_0, c_0 >$ if t is a partial matching-tree of t_0.

In the partial TSC $< t, s, c >$, if a leaf node n of t is a nonterminal node in t_0, then n is called a *substitution node*. The corresponding translation of n in the target-language string is replaced by the category symbol of the node. The category symbol in the target-language string is called *substitution symbol*. There exists a correspondence between the substitution node and the substitution symbol.

For a partial TSC $< t, s, c >$, s consists of both the target-language words and the substitution symbols. s is similarly determined as the string in maximal TSC except that the translation of the substitution node is replaced with the substitution symbol. In Fig. 2b, there are two substitution nodes, 6 and 11 (highlighted in bold). For the Chinese string, the symbols in the angle brackets are the substitution symbols. The first substitution symbol is the replacement of 她朋友*ta pengyou* 'her friend', which is the translation of the node 11. The second one is the replacement of 一本书*yi ben shu* 'a book', which is the translation of the node 6.

In the partial TSC $< t, s, c >$, c represents the correspondence between the leaf nodes of t and the words (or the substitution symbols) in s.

2.5. Homologous tsc

Given a TSC $< t_0, s_0, c_0 >$, TSC $< t_0, s_i, c_i >$ is called the *homologous TSC* of $< t_0, s_0, c_0 >$.

The matching-tree of a TSC is the same to the matching-tree of its homologous TSC. But the target-language strings and the correspondences in a TSC and its homologous TSC are different. "Two TSCs are Homologous" means that the same source-language tree can be translated into different target-language strings. For example, Fig. 3 shows the TSCs which are the homologous TSCs of the partial TSC in Fig. 2b.

2.6. Unaligned word in a tsc

In a bilingual sentence pair, some words in one language, such as quantifiers and some of the auxiliary words, have no counterparts in the other language. For example, in Fig. 1, the Chinese words 那里*nali* 'there' and 了*le* (an aspect marker), have no counterparts in the source English sentence.

However, these unaligned words are important to ensure the fluency of the translation. In this article, when combining the translation fragments, we deal with any unaligned words adjacent to translation fragments as follows.

Give a TSC $< t, s, c >$, s is first determined as described in Sects. 2.3 and 2.4. Then the unaligned words adjacent to s or substitution symbols are regarded as optional words. The statistical translation generation procedure, described in Sect. 4, will make a decision as to whether or not to keep these unaligned words in the final translation.

Figure 4 shows a TSC with a unaligned word, 那里*nali* 'there'.

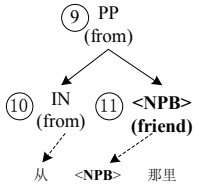


Figure 4. TSC with unaligned word

3. TSC-tree matching

A TSC $< t, s, c >$ matches a parse tree T, if the following conditions are satisfied:

1. The nonterminal node of t matches both the headword and the category of the corresponding node of T;
2. the leaf node of t matches at least the category of the corresponding node of T.

In the translation example database, there may be many TSCs that match an input parse tree. For each TSC, we assign a matching score to describe its similarity with the parse tree. Based on the matching score, we use a greedy tree-matching algorithm to search out the TSC forest which best matches the given parse tree.

3.1. Matching criterion

We have the following assumptions about the matching between a TSC and a tree:

1. The more nodes in the TSC, the better;
2. The higher the semantic similarity between the substitution node of TSC and its corresponding node of the parse tree, the better.

Therefore, the matching score between a TSC $< t, s, c >$ and a parse tree T is defined as the sum of the similarity between the nodes in t and their corresponding nodes in T. It is calculated as shown in (3).

$$M(< t, s, c >, T) = \sum_{n_i \in t} Sim(n_i, n'_i) \quad (3)$$

where, n_i is the i^{th} node in t, n'_i is the corresponding node of n_i in T, and $Sim(n_i, n'_i)$ is the similarity between n_i and n'_i.

If both the headword and the category of n_i are identical with that of n'_i, $Sim(n_i, n'_i)$ is set to 1.0. Otherwise, $Sim(n_i, n'_i)$ is defined as the semantic similarity between the headwords of the two corresponding nodes, as in (4),

(4) $Sim(n_i, n_i') = WordSim(f_i, f_i')$

where, f_i is the headword of n_i, f_i' is the headword of n_i'.

In this work, the semantic similarity between English words is calculated using WordNet (Fellbaum 1998). We employ the same method as described in Lin (1998) to calculate the semantic similarity between words in the semantic hierarchy, as shown in (5),

(5) $WordSim(f_1, f_2) = \frac{2 \times \log p(C_0)}{\log p(C_1) + \log p(C_2)}$

where C_i is the concept that f_i belongs to, C_0 is the nearest common ancestor in the semantic hierarchy that subsumes both C_1 and C_2, and $p(C_i)$ is the probability of encountering an instance of C_i in the corpus.

3.2. Matching algorithm

A TSC forest matching a parse tree means that the matching trees in the TSC forest can exactly compose the parse tree. For a parse tree, there may be many TSC forests that match it. We need to find the TSC forest that best matches the input tree. In all the forests matching the input tree, the TSC forest with the highest average matching score best matches the parse tree. We suppose that the best matching TSC forest will be used to generate the best translation.

We use a greedy matching algorithm to search for a matching TSC forest. The result of the greedy matching algorithm is good but not always optimal. However, the greedy algorithm balances the translation quality and translation efficiency. For a given parse tree, the algorithm first searches for the best matching TSC. Then, for each substitution node in this TSC, the algorithm searches for the TSC that best matches the subtree of the parse tree rooted at the substitution node. This procedure is iterated untill all substitution nodes are expanded. Figure 5 shows the outline of the algorithm.

Let F be a TSC forest found by the tree-matching algorithm, and $< t_i, s_i, c_i >, < t_j, s_j, c_j > \in F$. If the root node of t_j corresponds to a substitution node of t_i, then $< t_j, s_j, c_j >$ is called the *child TSC* of $< t_i, s_i, c_i >$, and $< t_i, s_i, c_i >$ is called the *parent TSC* of $< t_j, s_j, c_j >$.

Figures 6 and 7 show the tree-matching procedure. The input sentence is (6). Figure 6 shows the parse tree of the input sentence. Figure 7 shows the TSC forest that best matches the parse tree.

(6) The city is China's principal capital market.

The algorithm first searches for all TSCs that match the parse tree rooted at the node "TOP (is)". Among all the candidate TSCs, TSC (a) in Fig. 7 achieves the highest matching score and therefore is considered to be the best match of the parse tree.

In TSC (a), the substitution node "<VP>(is)" has a counterpart node "VP (is)" in the parse tree. Then the algorithm continues to search for the TSCs that match the matching tree rooted at "VP (is)" in the parse tree. As a result, TSC (b) in Fig. 7 achieves the highest matching score among all the candidate TSCs and is considered to be the best match. Similarly, the substitution node in TSC (b) has a counterpart node in the parse tree. So the algorithm searches for the TSC that best matches the matching tree rooted at the counterpart node in the parse tree. This procedure continues until there is no substitution node in the selected TSC. Finally, a TSC forest that includes five TSCs is obtained. In this forest, TSC (b) is rooted at the substitution node in TSC (a). Thus, TSC (b) is the child TSC of TSC (a) and TSC (a) is the parent TSC of TSC (b).

Input Parse tree T_{in}
  TSC forest F is set to NULL
  If *SearchTSC*(T_{in}, F) = TRUE
    Return F
  Else
    Return NULL
Output TSC forest F or NULL

Function *SearchTSC*(Matching-tree T, TSC forest F)
  Set n_0 to the root of T
  Search for all TSC matched T and rooted at n_0
  Set c to the number of searched TSCs
  If c = 0 return FALSE
  Sort the searched TSCs in descending order of the matching score
  Set i = 1
(1) For each substitution node n_k of TSC_i
  Set n_s to the corresponding node of n_k in T
  Set T_s to be the maximal matching-tree of T and T_s is rooted at n_s
  If *SearchTSC*(T_s, F) = FALSE
    Remove all the TSCs, which are matched T, from F
    if i >= c return FALSE
    i = i + 1
    Goto (1)
  Add TSC_i to F
  Return TRUE

Figure 5. Tree-matching algorithm

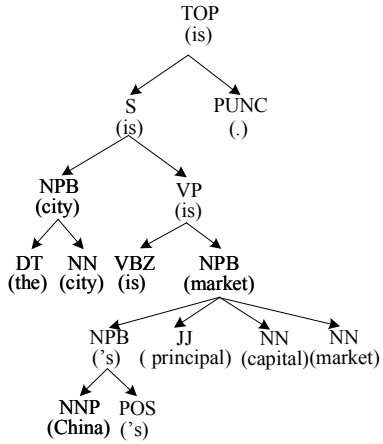


Figure 6. Example of tree-matching algorithm: Parse tree of the input sentence

4. Statistical generation

To generate the final translation, a statistical generation model is used to combine the target-language strings in the TSC forest in a bottom-up manner.

The following three feature functions are employed in statistical generation:

(1) The matching score between the tree in TSC and the input tree;

(2) The translation probability of translating the source word as the target word;

(3) The language-model probability for the target-language string.

Feature (1) and (2) measure the confidence of a TSC. Feature (3) measures the confidence of the target-language translation.

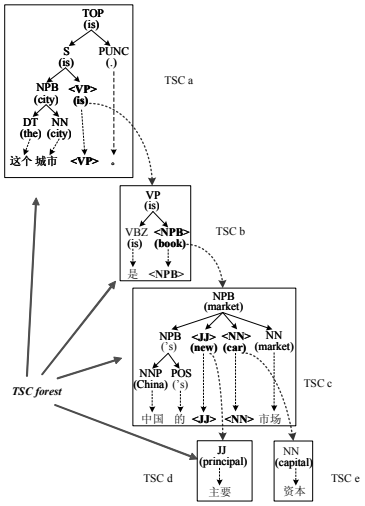


Figure 7. Example of tree-matching algorithm: TSC forest that best matches the parse tree

The final translation is determined by combining the feature functions as described in (7).

$$\tilde{t} = \text{argmax}_t\{\textstyle\sum_i \lambda_i \log(h_i)\} \tag{7}$$

where h_i is the feature function, λ_i is the weight of the feature function, and $\tilde{t}$ is the final translation.

To involve more possible translation candidates in the statistical generation procedure, we extend each TSC forest by adding its homologous TSCs in the forest.

4.1. Feature functions

The matching score is calculated as shown in (3) above. In (8), the matching score is denoted as $h_{\text{MS}}(< t, s, c >, T)$.

$$h_{\text{MS}}(< t, s, c >, T) = M(< t, s, c >, T) \tag{8}$$

The word-translation probability feature function measures the confidence of the word correspondence in the TSC. The word translation probability of a TSC is defined in (9),

$$h_{\text{TM}}(< t, s, c >) = \exp\{\textstyle\sum_{e_j \leftrightarrow f_i} \ln(p(e_j|f_i))/N\} \tag{9}$$

where $\exp(x) = e^x$, f_i is the headword of the leaf node in t, $e_j \leftrightarrow f_i$ denotes that e_j is the corresponding word of f_i in s, $p(e_j|f_i)$ is the probability of f_i translated to e_j, and N is the total number of leaf nodes in t.

The language-model probability feature function describes the fluency of a translation. A trigram language model is used to calculate the probability of a translation fragment occurring in the target language. The language model is described in (10),

$$h_{\text{LM}}(\mathbf{e}) = (\textstyle\prod_i p(e_i|e_{i-2}, e_{i-1}))^{1/n} \tag{10}$$

where $\mathbf{e}$ is the produced translation fragment produced by combining the target-language strings of TSCs, e_i is the i^{th} word in $\mathbf{e}$, and n is the length of the translation fragment. This parameter is introduced to prevent a preference for short translation fragments.

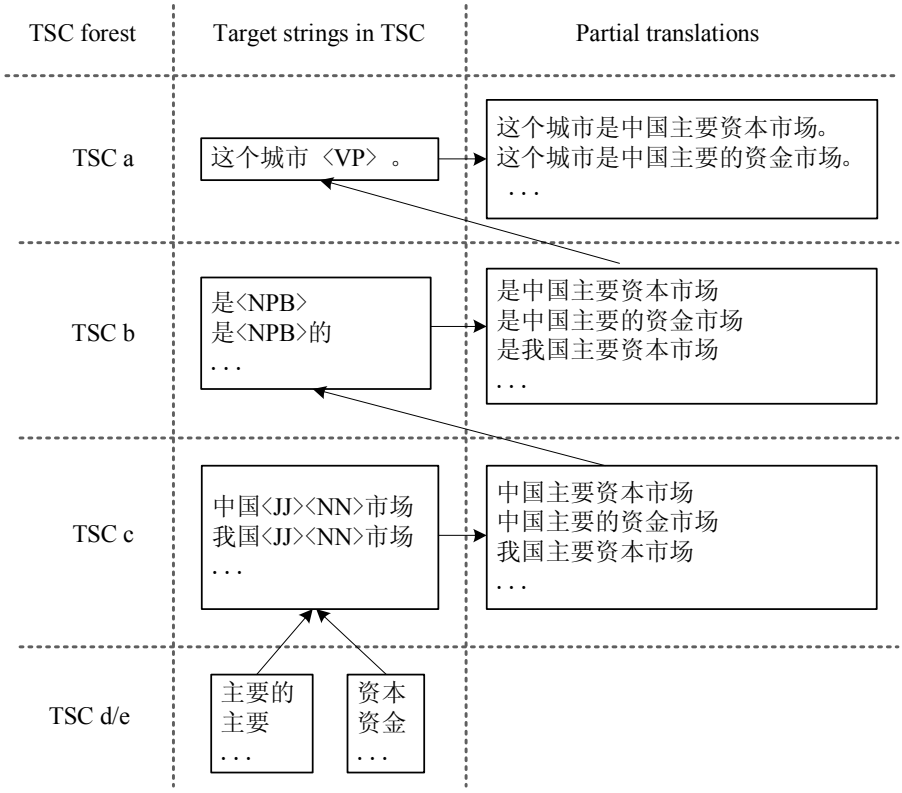


Figure 8. Example of translation generation

4.2. Bottom-up generation

The final translation is generated by combining the target-language strings in the TSC forest in a bottom-up manner (Imamura *et al.*, 2004). For a TSC $< t, s, c >$ in the forest, the n-best translation candidates are chosen by using the feature functions described in Sect. 4.1. If the target-language string s contains substitution symbol, then the symbol should be replaced with the translation of the child TSC of $< t, s, c >$.

Let TSC_0 be the TSC whose root node matches the root node of the parse tree. The best translation candidate of TSC_0 is regarded as the final translation of the input sentence.

Figure 8 shows the bottom-up generation for the TSC forest in Fig. 7. TSCs (d) and (e) do not contain any substitution symbol. The translation of TSCs (d) and (e) are the strings in the target language, respectively. TSC (c) contains two substitution symbols, and the translation is generated by replacing the substitution symbol <JJ> with the translation of TSC (d) and replacing the substitution symbol <NN> with the translation of TSC (e). In the same way, for TSC (a), the translation is obtained by replacing the substitution symbol <VP> with the translation of TSC (b). Finally, the best translation candidate of TSC (a) is regarded as the translation of the input sentence.

5. Resources used in experiments

The resources used in our experiments are translation examples, a test set, a translation dictionary, and a language model.

The translation examples include 262,060 English-Chinese bilingual sentence pairs collected from general language texts. The average length of the English sentences is 12.1 words while the average length of the Chinese sentences is 12.5 words. The source-language sentence is parsed using the

parser of Collins (1999). In the original result of the parser, the punctuation node always occurs as a right sibling of the previous leaf node. In this case, the punctuation node cannot always act as a coordinating conjunction (Bikel 2004). Thus, we move the position of the punctuation node in the tree. For a punctuation node n, if it is the leftmost or rightmost leaf node, then we set the root node of the tree as the parent node of n. Otherwise, let n_r be the nearest right neighbor of n. Then the nearest common ancestor of n and n_r is set as the parent node of n. The English and Chinese words are aligned using Giza++ (Och and Ney 2000).

The test set contains 500 English sentences which are not included in the translation examples. Each sentence in the test set has two translation references.

In this work, an English-Chinese translation dictionary is used, in two ways. One is to translate the words that cannot be translated using the translation examples. The other is to provide the word-translation probability for the English-Chinese word pair. The dictionary contains about 90,000 entries. Each entry is assigned a translation probability. The translation probability is calculated from the word-aligned sentence pairs in the translation examples, as defined in (11),

$$p(e|f) = \frac{C(e,f)}{C(f)} \tag{11}$$

where f and e represent the source and the target words, respectively, $C(f)$ is the frequency of occurrence of f, and $C(e, f)$ is the co-occurrence frequency of e and f.

The Chinese language model in our system is a standard trigram model. The language model is trained on a Chinese corpus in the general domain using the SRILM toolkit (Stolcke, 2002). The training data includes 228 m Chinese words. The perplexity of the language model with respect to the Chinese sentences in the translation examples is 31.85.

6. Results and evaluation

In this section, we describe five experiments which we performed to evaluate the translation generation components of our English-to-Chinese EBMT system.We first evaluate the generation components of our system, and then compare our system with statistical MT (SMT) systems. The NIST score (Doddington 2002) is used for evaluation.

6.1. Component evaluation

The five experiments carried out where as follows:

LM This model uses only the language model to score the generated translation. During the generation, for each matching tree in the TSC forest, the generated translation candidates are ranked according to the language-model probability.

MS This model employs only matching score to select translation fragments for the matching trees in the TSC forest. The final translation is obtained by combining the target-language strings of TSCs with the highest matching score.

LM+MS In this generation model, we combine the target-language model and the matching score to select translation fragments and to generate the final translation. In this model, we set $\lambda_{\mathrm{LM}} = 0.5$ and $\lambda_{\mathrm{MS}} = 0.5$.

LM+TM+MS This generation model is built by combining the three feature functions described in Sect. 4.1: (a) the language-model probability of the generated translation; (b) the word-translation model between the source part and the target part in TSC; (c) the matching score of

Table I. Experiment results of evaluating translation generation component

Experiments	NIST
LM	4.0053
MS	4.2803
LM + MS	4.7545
LM + TM + MS	4.8174
LM + TM + MS + UW	5.2577

the TSC. The weights of the feature functions are tuned with a development set which contains 200 English sentences, each of which has two translation references. We set $\lambda_{\mathrm{LM}} = 0.29$, $\lambda_{\mathrm{TM}} = 0.42$ and $\lambda_{\mathrm{MS}} = 0.29$. With these weights, we get the highest NIST score on the development set. During generation, the generated translations are ranked according to the combined score of the above three features.

LM+TM+MS+UW In the above four methods, unaligned words adjacent to translation fragments are not taken into account when constructing TSCs. In this method, besides the three feature functions in "**LM+TM+MS**", the unaligned words are taken into account. During generation, the statistical generation model determines whether these unaligned words should be added to the final translation or not.

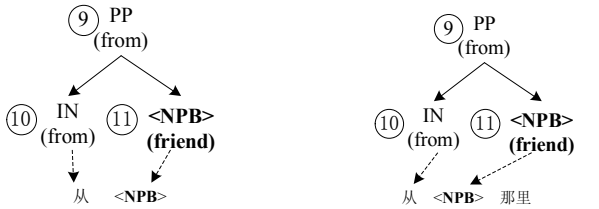
The experimental results are shown in Table I. From the results, it can be seen that "**TM+LM +MS+UW**" achieves the best translation result. This indicates that our translation generation method is very effective to improve translation quality.

"**LM+MS**" achieves much better translation quality than the two separate methods "**LM**" and "**MS**". When we incorporate the word-translation probability into "**LM+MS**", the translation quality is further improved. "**LM+TM+MS**" achieves a NIST score of 4.8174, which is higher than "**LM+MS**". This is because the matching score and the word-translation probability select translation examples from different aspects. The contribution of the matching score is to find a similar example to the input sentence in terms of both syntactic and semantic structures, while the word-translation probability improves target-word/phrase selection. Therefore, after introducing the above two features, the translation quality is unsurprisingly improved.

By introducing the unaligned words into the generation model, the translation quality of "**LM +TM+MS+UW**" is higher than that of "**LM+MS+TM**". In "**LM+TM+MS+UW**", the language model determines whether the unaligned words should be added to the final translation. In fact, adding the unaligned words improves the fluency of the translation. In order to explain this procedure, we construct two TSCs from the translation example in Fig. 1, which are rooted at the same node 9. One ignores the unaligned Chinese word 那里*nali* 'there', while the other includes it. The TSCs are shown in Fig. 9.

Both TSC(a) and TSC(b) in Fig. 9 are applied to translate the two input sentences 1 and 2 in Table II. The two translations shown in each case use TSC(a) or TSC(b), respectively, as indicated. The language-model score for each translation is calculated as $-\log(P_{\mathrm{LM}})$: the larger the value, the worse the translation.

From the results, it can be seen that (12a) is more fluent than (12b) but (13b) is more fluent than (13a). Thus, TSC(a) should be selected to translate sentence (12) while TSC(b) should be selected to translate sentence (13).



a. TSC without '那里' b. TSC with '那里'

Figure 9. TSCs w/o unaligned words

Table II. Example translations with the TSCs in figure 9

		Used TSC	-log(P_{LM})
Input 1	He got down from the bus.		
T11	他 从 公共汽车 下来 。 *Ta cong gonggongqiche xialai* HE FROM BUS GET DOWN	(a)	10.98
T12	他 从 公共汽车 那里 下来 。 *Ta cong gonggongqiche nali xialai* HE FROM BUS THERE GET DOWN	(b)	14.77
Input 2	He got the alms from the government.		
T21	他 从 政府 得到 救助金 。 *Ta cong zhengfu dedao jiuzhujin* HE FROM GOVERNMENT GET ALMS	(a)	18.36
T22	他 从 政府 那里 得到 救助金 。 *Ta cong zhengfu nali dedao jiuzhujin* HE FROM GOVERNMENT THERE GET ALMS	(b)	16.39

In our system, the two TSCs in Fig. 9 are represented as one TSC. The unaligned word 那里*nali* 'there' is retained in the target-language string of the TSC as an optional word. As the examples shows, the statistical generation model determines whether the unaligned word is added to the translation based on the contexts. As a result, the fluency of the translation is improved.

6.2. Comparisons with SMT

Besides the above evaluation, we also compared our system with two SMT systems, as described in the following paragraphs.

The ISI ReWrite decoder (Germann 2003) is a *word-based* SMT system. The decoder supports IBM Model 4 (Brown et al. 1993) using the translation model and language model. The decoder implements a greedy but efficient decoding algorithm to translate the source sentence into the target sentence. We used Giza++ to train the parameters of the translation model. The training data of the translation model were the same translation examples as used in our EBMT system, and we used the same language model as in our system.

Pharaoh (Koehn et al. 2003; Koehn 2004) is a *phrase-based* SMT decoder. We used the default feature set: language model, reordering model, phrase translation table, and word penalty. We ran the trainer with its default settings (maximum phrase length 7) and then use Koehn's implementation of minimum error-rate training (Och 2003) to tune the feature weights on our development set. Again,

Table III. Evaluation results of the SMT systems and our system

Systems	NIST
Word-based SMT System	4.5239
Phrase-based SMT System	5.2214
Our System	5.2577

the training data of the translation model were the same translation examples as used in our EBMT system, and we used the same language model as in our system.

We evaluated the SMT systems using the same test set as described in Sect. 5.1. Table III shows the results of the SMT systems together with those for our system. From the results, it can be seen that our system achieves a much higher NIST score than the word-based SMT system, and a slightly higher NIST score than the phrase-based SMT system. These results indicate that the performance of our system is at least comparable with that of the phrase-based SMT system.

7. Conclusion and future work

This article proposed an EBMT method based on TSCs and statistical generation. According to the proposed EBMT method, the translation examples are represented as TSCs, which are obtained only by parsing the source sentences and aligning the source and target words in the translation examples. The TSC is a triple of a parse tree in the source language, a string in the target language, and the correspondence between the leaf node of the source-language tree and the substring of target-language string. A greedy TSC tree-matching algorithm is proposed to find the TSC forest which best matches the input parse tree efficiently. The generationmodel combines three feature functions: the matching score of the TSC, the translation probability of translating the source word as the target word, and the target-language model. Moreover, in order to ensure the fluency of the generated translation, we also introduce any unaligned words adjacent to translation fragments into the TSC. The generation model is used to determine whether these unaligned words are added to the final translation. Component evaluation indicates that the three feature functions in the generation model effectively improve the translation fragment selection and combination. It also shows that introducing unaligned words into TSCs increases the fluency of the generated translation. In addition, we compared our system with word- and phrase-based SMT systems. The results indicate that the translation quality of our system is comparable with state-of-the-art phrase-based SMT systems.

In future work, we expect that better translation could be achieved by adding more features to the generation model.We will also try to optimize the weights in the generation model, using methods such as minimum error-rate training described in Och (2003). Furthermore, we will investigate the possibility of leveraging our TSC method into the SMT framework.

References

Yasuhiro Akiba, Taro Watanabe and Eiichiro Sumita (2002) Using Language and Translation Models to Select the Best among Outputs from Multiple MT systems. In: the 19th International Conference on Computational Linguistics, Taipei, Taiwan, pp. 8-14. August 2002

Mosleh H. Al-Adhaileh and Tang Enya Kong (1999) Example-Based Machine Translation Based on the Synchronous SSTC Annotation Schema. In: Machine Translation Summit VII, Singapore, pp. 244-249. September 1999

Mosleh H. Al-Adhaileh, Tang Enya Kong and Zaharin Yusoff (2002) A Synchronization Structure of SSTC and its Applications in Machine Translation. In: the Coling-2002 Post-Conference Workshop on Machine Translation in Asia, Taipei, Taiwan, pp. 1-8. August 2002

Eiji Aramaki and Sadao Kurohashi (2004) Example-Based Machine Translation Using Structural Translation Examples. In: International Workshop on Spoken Language Translation, Kyoto, Japan, pp. 91-94. September 2004

Eiji Aramaki, Sadao Kurohashi, Hideki Kashioka and Hideki Tanaka (2003) Word Selection for EBMT based on Monolingual Similarity and Translation Confidence In: the HLT/NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond, Edmonton, Canada, pp. 57-64. May 2003

Daniel M. Bikel (2004) Intricacies in Collins' Parsing Model. Computational Linguistics 30 (4): 479-511

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra and Robert L. Mercer (1993) The Mathematics of Machine Translation: Parameter Estimation. Computational Linguistics 19(2): 263-311

Chris Callison-Burch and Raymond S. Flournoy (2001) A Program for Automatically Selecting the Best Output from Multiple Machine Translation Engines. In: Machine Translation Summit VIII, Santiago de Compostela, Spain, pp. 63-66. September 2001

Michael Collins (1999) Head-Driven Statistical Models for Natural Language Parsing. Dissertation, University of Pennsylvania

George Doddington (2002) Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. In: the 2th Conference on Human Language Technology, San Diego, CA, pp. 128-132. March 2002

Ulrich Germann (2003) Greedy Decoding for Statistical Machine Translation in Almost Linear Time. In: Human Language Technology and North American Association for Computational Linguistics Conference, Edmonton, Canada, pp. 72-79. May 2003

Kenji Imamura, Hideo Okuma, Taro Watanabe and Eiichiro Sumita (2004) Example-Based Machine Translation Based on Syntactic Transfer with Statistical Models. In: the 20th International Conference on Computational Linguistics, Geneva, Switzerland, pp. 99-105. August 2004

Satoshi Kaki, Setsuo Yamada and Eiichiro Sumita (1999) Scoring Multiple Translations Using Character N-gram. In: the 5th Natural Language Processing Pacific Rim Symposium, Beijing, China, pp. 298-302. November 1999

Dekang Lin (1998) An Information-Theoretic Definition of Similarity. In: the 15th International Conference on Machine Learning, San Francisco, CA, pp. 296-304. July 1998

Philipp Koehn, Franz Josef Och and Daniel Marcu (2003) Statistical Phrase-Based Translation. In: Human Language Technology and North American Association for Computational Linguistics Conference, Edmonton, Canada, pp. 127-133. May 2003

Philipp Koehn (2004) Pharaoh: A Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In: the Sixth Conference of the Association for Machine Translation in the Americas, Washington DC, America, pp. 115-124. September 2004

Yuji Matsumoto, Hiroyuki Ishimoto and Takehito Utsuro (1993) Structural Matching of Parallel Texts. In: the 31st Annual Meeting of the Association for Computational Linguistics, Columbus, Ohio, pp. 23-30. June 1993

Franz Josef Och (2003) Minimum Error Rate Training in Statistical Machine Translation. In: the 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan, pp. 160-167. July 2003

Arjen Poutsma (2000) Data-Oriented Translation. In: the 18th International Conference on Computational Linguistics, Saarbrücken, Germany, pp. 635-541. August 20

Stuart M. Shieber (1994) Restricting the Weak Generative Capacity of Synchronous Tree Adjoining Grammar. Computational Intelligence 10(4): 371-385

Harold Somers (1999) Review Article: Example-based Machine Translation. Machine Translation 14(2): 113-157

Andreas Stolcke (2002) SRILM - An Extensible Language Modeling Toolkit. In: the International Conference on Spoken Language Processing, Denver, Colorado, pp. 901-904. September 2002

Takehito Utsuro, Kiyotaka Uchimoto, Mitsutaka Matsumoto and Makoto Nagao (1994) Thesaurus-Based Efficient Example Retrieval by Generating Retrieval Queries from Similarities. In: the 15th International Conference on Computational Linguistics, Kyoto, Japan, pp. 1044-1048. August 1994

Hideo Watanabe (1992) A Similarity-Driven Transfer System. In: the 14th International Conference on Computational Linguistics, Nantes, France, pp. 770-776. August 1992

Hideo Watanabe (1995) A Model of a Bi-Directional Transfer Mechanism Using Rule Combinations. Machine Translation 10(4): 269-291

Andy Way (2003) Machine Translation Using LFG-DOP. In: R. Bod, R. Scha & K. Sima'an (ed) Data-Oriented Parsing, CSLI Publications, Stanford, CA., pp. 359-384