A New Translation Template Learning Based on Hidden Markov Modeling

NGUYEN MINH LE, AKIRA SHIMAZU, and SUSUMU HORIGUCHI
Graduate School of Information Science, JAIST
ISHIKAWA 923-1292, JAPAN

Abstract: This paper addresses a novel translation method based on Hidden Markov Model, using template rules that are learned from the bilingual corpus. The method can enhance the translation accuracy and ensure a low complexity in comparing with previous template learning translation method, and presents a new perspective for applying statistical machine learning on example based translation domain.

Keywords: Machine translation, EBMT, Translation Template learning, HMM.

1 Introduction

Example based translation (EBMT), originally proposed by Nagao [1], is one of the main approaches of corpus-based machine translation. Following Nagao's original proposal, several methods were presented by Nagao and Sato [2], Sumita and Iida [3], Niren Burg et al[4] and Brown [5]. The excellent review paper of EBMT [6] describes the main idea behind EBMT as follows. A given input sentence in the source language is compared with the example translations in the given bilingual parallel text to find the closest matching examples so that these examples can be used in the translation of that input sentence. After finding the closest matching for the sentence in the source language, parts of the corresponding target language sentence are constructed using structural equivalences and deviances in those matches.

Cicekli and Güvercin [7][8] proposed a learning translation template which was applied the Nagao’s approach to translations from English to Turkish. This method is one of the successful methods and uses the similarity and difference between a source sentence and a target sentence in the given bilingual corpus to build template rules for translation. Its advantage is that it does not need complex syntactic or semantic parsing and it overcomes the imperfections of rule-based machine translation. One of the disadvantages of the method is that many templates can be matched with a particular input sentence. To overcome this problem, Öz [9] present a method which allows template rules to sort the translation results according to their confidence factors. However, in order to obtain the output results, this method needs to evaluate all matching rules for each input sentence, and many of these are redundant rules. The exponential calculation problem will arise when an input sentence is long and the number of template rules is large.

Here, we present a novel method based on an HMM model that uses constraints to establish a set of matching rules for each input sentence. Thus, we can avoid the exponential calculation problem and find the best results for translation an input sentence by performing a dynamic algorithm on HMM model.

The remainder of this paper is organized as follows: The template learning algorithm is given in Section 2. Section 3 describes an HMM model for translation using template rules. Section 4 shows experiments on an English-Vietnamese translation system, and Section 5 presents our conclusions and discusses some outstanding problems to be solved in the future work.

2 Translation Template Learning

The Translation Template Learning algorithm (TTL) infers translation templates using similarities and differences between two translation examples, \( E_a \) and \( E_b \), taken from a bilingual parallel corpus. Formally, a translation example \( E_a: E_a^1 \leftrightarrow E_a^2 \) is
composed of a pair of sentences, $E_a^1$ and $E_a^2$, that are translations of each other in English and Vietnamese respectively. A similarity between two sentences of a language is a non-empty sequence of common items (root words or morphemes) in both sentences. A difference between two sentences of a language is a pair of two sequences $(D_1, D_2)$ where $D_1$ is a sub-sequence of the first sentence, $D_2$ is a sub-sequence of the second sentence, and $D_1$ and $D_2$ do not contain any common items. Given two translation examples ($E_a, E_b$), we try to find similarities between the constituents of $E_a$ and $E_b$. A sentence is considered as a sequence of lexical items. If no similarities can be found, then no template is learned from these examples. If there are similar constituents then a match sequence $M_{a,b}$ in the following form is generated.

$$S_1^1, D_1^1, S_2^1, ... , D_{m-1}^1, S_n^1$$

for $1 \leq n, m$

Here, $S_k^1$ represents a similarity (a sequence of common items) between $E_a^1$ and $E_b^1$. Similarly, $D_k^1 : (D_{k,a}^1, D_{k,b}^1)$ represents differences between $E_a^1$ and $E_b^1$, where $D_{k,a}^1, D_{k,b}^1$ are non-empty different items between two similar constituents $S_k^1, S_{k+1}^1$.

For instance, let us assume that the following translation examples are given:

“*I bought the book for John*” $\leftrightarrow$ “Tôi đã mua một quyển sách cho John”

“*I bought the ring for John*” $\leftrightarrow$ “Tôi đã mua một chiếc nhẫn cho John”

For these translation examples, the matching algorithm obtains the following match sequence.

$I$ bought the (book, ring) for John $\leftrightarrow$ Tôi đã mua một (quyển sách, chiếc nhẫn) cho John

That is, $S_1^1 = I$ bought the, $D_1^1 = ($book, ring$), S_1^2 = $ for John, $S_2^1 =$Tôi đã mua một, $D_2^1 = ($quyển sách, chiếc nhẫn$), S_2^2 = $cho John$.$

After a match sequence is found for two translation examples, we use two different learning heuristics to infer translation templates [7][8] from that match sequence. These two learning heuristics try to locate corresponding differences or similarities in the match sequence. The first heuristic, the Similarity Translation Template Learning algorithm (STTL), tries to locate all corresponding differences and generates a new translation template by replacing all of the differences with variables. The second heuristic, the Difference Translation Template Learning algorithm (DTTL), can infer translation templates by replacing similarities with variables, if it can locate corresponding similarities in the match sequence. The STTL and DTTL are combined as the Translation Template Learning algorithm (TTL). From the corpus, the TTL algorithm tries to infer translation templates using the above two algorithms. After all of the translation templates have been learned, they are sorted according to their specificities. Given two templates, the one that has a higher number of terminals is more specific than the other.

In the following section, we address a new method to increase translation accuracy and reduce calculation complexity.

### 3 Translation Template Learning Base on HMM

To explain translation template learning based on HMM model, some notations are defined below, followed by the presentation of a translation based HMM model.

#### 3.1 Template Rules

Let $SL$ and $TL$ be the source and target languages and $S_1S_2...S_n \leftrightarrow T_1T_2...T_k$ be template rules, in which $S_i$ is a sequence of word or a variable in $SL$ and $T_i$ is a sequence of words (called a constant element) or a variable in $TL$. Each variable in the left side is aligned with each variable in the right. A variable in the left side and a variable in the right side of a template rule will be received as a phrase or a word in SL and TL, respectively. Figure 1 depicts an example of template rules in which a sentence containing “give...up” in English is translated to a sentence in Vietnamese containing “tu bo”.

```
S_1       S_1
    give
    up

T_1      T_1
    tu bo
```

Figure 1. Template rule example

Let a *lexical rule* be a template rule that has no variable inside. A *lexical rule* is a bilingual phrase in SL and TL.
3.2 Translation base on HMM Model

3.2.1 The Model

The model we propose has two steps. First, we formulate template learning translation as an equivalent problem that can be solved by using the HMM model, based on a set of constraints rules which are observed from the characteristics of SL and TL and on a training corpus. Next, a dynamic programming technique, a variant of the Viterbi algorithm, is used to find the best translation results.

Problem: Given an input sentence \( e_1 e_2 \ldots e_m \) (\( e_i \) is a word) and a set of template rules \( r_1, r_2, \ldots, r_d \), find the set of rules so that their translation results best explain that sentence. For convenience we will use \( e[1: m] \) as shorthand for the input sentence \( e_1 e_2 \ldots e_m \). The problem is equivalent to finding all translation results for each rule \( r_i \) (i=1, d).

Assuming that the rule \( r_i \) is defined as \( S_1 S_2 \ldots S_n \leftrightarrow T_1 T_2 \ldots T_k \), the original method [8] tries to find all possible ways to replace the variables with phrases in SL so that the input sentence \( e[1:m] \) can be produced from this rule. Next, it finds each corresponding phrase in TL within the set of lexical rules with a phrase in SL, in order to transform the input sentence into the target language. However, when the input sentence is long and the number of rules is large with many variables inside, the original method has to cope with an exponential calculation.

To overcome this problem, we propose an approach based on HMM modeling, as discussed below. Figure 2 shows that an input sentence can be decomposed in many ways using the left side of the template rules. Suppose that the variables \( X \) and \( Y \) each have 10 elements whose substrings can be found in the input sentence on the left sides of the lexical rules. For each element in the variable \( X \) which has a position \( k \) within the input sentence we have to find all elements for the variable \( Y \) that have substrings which start from a position \( k+1 \) and are also on the left side of a lexical rule. Thus, we have to consider \( 10 \times 10 \) translation combinations, most of which must be discarded. From the example in Figure 2, each constant, \( s_j \) can be associated with a phrase in the right side of the rule, \( r_i \), and each variable \( s_j \) within the rule \( r_i \) can be associated with a set of lexical rules whose left side is a substring that could start from many possible positions within the input sentence.

In such a framework, we can assume that a lexical rule corresponds to a hidden state and a substring in the input sentence to an observed symbol produced from the state, and that the problem of translation is equivalent to finding a lexical rule for each variable.

To find the sequence of lexical rules that maximizes formula (3), a kind of dynamic programming, the Viterbi algorithm [10] can be used. If the rule \( r_j \) has \( n \) variables and each variable consists of \( l \) elements, then the complexity is \( n \times l^n \), while the recursive way is \( l^n \). In addition, each rule \( r_j \) can be assigned a translation score as the value of formula (3) and output translations for the input sentence can be sorted according to the score value on the whole set of template rules. Therefore, using HMM modeling enables us to avoid the exponential calculation problem by using the dynamic Viterbi algorithm. In addition, it can sort translation results with higher accuracy without the need for complex processing on a set of template rules. It also presents a new perspective for applying statistical machine
learning theories in the example-based translation domain.

3.2.2 Estimation of HMM Model
The HMM model for translation is estimated by using the Forward-Backward learning [11]. The corpus of source sentences and target sentences is used to generate observed sequences. Each source sentence is translated by using a sequence of lexical rules if the right-hand side of the rules is the same as the target sentence within the corpus. After obtaining a sequence of lexical rules, the sequence of observed symbols is generated because each observed symbol is a left-hand side of a lexical rule. Therefore, using a set of template rules and the corpus we can generate a training data as follows:

\[ O_{i_1}O_{i_1+1}...O_{i_n} \Leftrightarrow S_{i_1}S_{i_1+1}...S_{i_m} \]
\[ O_{j_1}O_{j_1+1}...O_{j_n} \Leftrightarrow S_{j_1}S_{j_1+1}...S_{j_m} \]
\[ ... \]
\[ O_{k_1}O_{k_1+1}...O_{k_n} \Leftrightarrow S_{k_1}S_{k_1+1}...S_{k_m} \]

Here, \( O_{i_1}O_{i_1+1}...O_{i_n} \) is a sequence of observed symbols, \( S_{i_1}S_{i_1+1}...S_{i_m} \) is a sequence of lexical rules, and \( O_{j_1}O_{j_1+1}...O_{j_n} \Leftrightarrow S_{j_1}S_{j_1+1}...S_{j_m} \) means that a sequence of observed symbols is associated with the sequence of lexical rules. Suppose that \( c(l^i), c(l^i', l^i) \) and \( c(o^j, l^i) \) are the number of occurrences of lexical rule \( l^i \), the number of occurrence of lexical rule \( l^i' \) following lexical rule \( l^i \), and the number of occurrences of an observed symbol \( o^j \) corresponding with a lexical rule \( l^i \) respectively. With these notations, the initialization algorithm for estimating an HMM model by performing the Forward-Backward algorithm on the training data above is described as follows:

```plaintext
Figure 3. Algorithm for initializing the parameters of the HMM model for template rules
```

After initializing the probability of observed symbols and lexical rules, Forward-Backward learning is used to estimate the HMM for translation.

3.2.3 Example
We describe a translation example using the original method and the HMM method for an input sentence with a template rule and a set of lexical rules as shown in the Table 1.

There are three translation outputs when applying the original method:

\( (L_1,L_2,L_3) \), \( (L_1,L_4,L_5) \), \( (L_1,L_6,L_7) \).

Suppose that the probabilities of two lexical rules in the example are estimated as follows:

\[ P(L_1|L_2)=0.2; \quad P(L_1|L_4)=0.6; \quad P(L_1|L_6)=0.2; \]
\[ P(L_2|L_3)=0.2; \quad P(L_4|L_5)=0.5; \quad P(L_6|L_7)=0.2. \]

Using formula (3), we have \( P(L_1,L_2,L_3)=0.3, P(L_1,L_6,L_7)=0.04 \).

Thus, the translation result is the likely sequence of lexical rules \( (L_1,L_4,L_5) \).

<table>
<thead>
<tr>
<th>Table 1. An example of translation using template translation learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: I do not think it is necessary to launch a full inquiry at this time</td>
</tr>
<tr>
<td>Human translation: Tôi không nghĩ là nó thực sự cần thiết để bắt đầu cuộc điều tra ở thời điểm này</td>
</tr>
<tr>
<td>EBMT (the original algorithm has to enumerate all translation results)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lexical rule</th>
<th>Template rule: X “necessary to launch” Y Z ↔ X “cần thiết để bắt đầu” Y’ Z’</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>I do not think it is ↔ tôi không nghĩ nó là</td>
</tr>
<tr>
<td>L2</td>
<td>a full ↔ sự đầy đủ</td>
</tr>
<tr>
<td>L3</td>
<td>inquiry at this time ↔ đòi hỏi ở thời điểm này</td>
</tr>
<tr>
<td>L4</td>
<td>a full inquiry ↔ một cuộc điều tra đầy đủ</td>
</tr>
<tr>
<td>L5</td>
<td>at this time ↔ ở thời điểm này</td>
</tr>
<tr>
<td>L6</td>
<td>a full inquiry at ↔ một cầu hỏi đầy đủ ở</td>
</tr>
<tr>
<td>L7</td>
<td>this time ↔ thời gian này</td>
</tr>
</tbody>
</table>

Table 1 shows that the original method has to enumerate all translation results, while the proposed method can obtain the best translation results by applying a dynamic algorithm.

4 Experiments and Discussion
In order to assert our method can enhance the accuracy in translation while ensure the complexity
is low. We implemented an English Vietnamese translation and tested it on a corpus of 1200 bilingual sentences collected manually from some text books and newspapers. We are experimenting on the HMM model.

4.1 Template Translation Learning

Figure 4 shows the number of template translation rules vs. the number of sentences within the corpus. These results show how the number of template rules for a bilingual English-Vietnamese corpus increase with the number of sentences.

![Figure 4. The relation of the number of lexical rules and the number of template rules to the number of sentences within the corpus.](image)

The number of sentences in the corpus is from 300 to 1200 sentences. The solid line and the dotted line show the relation between the number of template rules and the number of lexical rules with the number of sentences within the corpus.

4.2 Constraints Application

In case of the corpus size is small, we can obtain a HMM model by using constraint application. We simplify the model by using constrains instead of the probability of Bigram and get translations using the Viterbi algorithm.

Let A and B be two lexical rules. The constraints are as follows:

**Constraint 1:** A sequence of words translated by the lexical rule A and B is permitted if the right sides of A and B satisfy the morphological condition in the target language and they are two consecutive positions in the output translation.

**Constraint 2:** A sequence of words translated by the lexical rule A and B is permitted if there exits a Vietnamese sentence within the corpus that contain two right sides of A and B.

4.3 HMM Model

In our example, there are 11,034 template rules and 2,287 lexical rules, using the template translation learning. The number of lexical rules is the number of hidden states in our HMM model. Using the template rules and the data corpus, we obtained the training data for estimating the HMM model described in section 3.2.2; the initialization parameters for the HMM model were estimated using the algorithm in Figure 3. The training data for estimating the HMM model consists of 1200 observed sequences; each sequence corresponds to a sequence of lexical rules. We used 1100 observed sequences to initialize the parameters for HMM models, using the algorithm in Figure 3. Afterward, the Forward and Backward algorithm was applied to the remaining sequences to train the model.

4.4 Experimental Results

After we generated a set of template rules on the corpus, we estimated the HMM model as described above. We tested the translation accuracy by using the sentences within the corpus.

Using the Viterbi algorithm for each rule, we were able to obtain a list of output translations. We compared the translation results from our methods with those of the original method by calculating correct translations in the total translation outputs. We obtained the results shown in Table 2. The sentences within the corpus were selected randomly and used as inputs for both the original method and our methods.

Table 2 shows that the constraints application and the translation based-HMM achieved better results in comparison with the original TTL algorithm. In addition, our method achieved a lower complexity, $O(n \times l^2)$, in comparison with the original method $O(l^*)$, in which $l$ is the number of lexical rules and $n$ is number of variables in a template rule. This was due to our use of a dynamic algorithm to avoid the exponential calculation problem.

<table>
<thead>
<tr>
<th>Table 2. Performance results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage of correct results by the original method</td>
</tr>
<tr>
<td>48%</td>
</tr>
</tbody>
</table>
Some examples of our translation method are described in Table 3. The second column of the table shows the best translation results achieved by our methods.

Table 3. Some examples of our translation results

<table>
<thead>
<tr>
<th>Input sentence</th>
<th>Translation output</th>
</tr>
</thead>
<tbody>
<tr>
<td>How long will you stay here?</td>
<td>Anh sẽ ở lại đây được bao lâu?</td>
</tr>
<tr>
<td>My book is as interesting as yours</td>
<td>Quyết sách của tôi thì lý thú ngang với quyết sách của anh</td>
</tr>
<tr>
<td>Several new proposals are being considered by the committee</td>
<td>Nhiều dự án mới đang được uy ban cứu xét</td>
</tr>
<tr>
<td>Before long rice seedlings were big enough to be planted in the field</td>
<td>Chăng bao lâu sao các cây lúa đó đủ lớn để được cây vào ruộng được</td>
</tr>
<tr>
<td>Have you written your report yet?</td>
<td>Anh viết xong bản báo cáo chưa?</td>
</tr>
<tr>
<td>If she had seen the movie, she would have told you</td>
<td>Nếu cô ta đã nhìn thấy phim, cô ta đã nói với bạn</td>
</tr>
</tbody>
</table>

4.5 Discussion

In case of corpus size is small, we are able to obtained HMM model by using constraints application since our method doesn’t depend on the size of corpus. When the corpus size is large enough, the HMM model can be estimated by using a common algorithm such as the forward-backward algorithm.

In both cases, the translation template learning using HMM model significantly improved the accuracy and the computation calculation in comparing with the original algorithm.

5 Conclusion

Our use of HMM modeling avoids the exponential calculation problem by using a dynamic algorithm. In addition, it can sort translation results according to the better accuracy without any complex process in set of template rules and therefore, ensure the translation accuracy. Moreover, it draws a new perspective for applying statistical machine learning theory on example based translation domain. Merging our proposed method with rule-based translation method is currently underway.

Acknowledgment

We would like to thank to Judith Steeh for editing the paper. This research was supported in part by the JAIST international research project grant and JSPS Grant-in-Aid for Scientific Research.

References