An Example-based Decoder for Spoken Language Machine Translation

Zhou-Jun Li
National Laboratory for Parallel and Distributed Processing, Changsha, China
School of Computer Science and Engineering, Beihang University, China
lizj@buaa.edu.cn

Wen-Han Chao
National Laboratory for Parallel and Distributed Processing, Changsha, China
cwh2k@163.com

Yue-Xin Chen
National Laboratory for Parallel and Distributed Processing, Changsha, China

Abstract

In this paper, we propose an example-based decoder for a statistical machine translation (SMT) system, which is used for spoken language machine translation. In this way, it will help to solve the re-ordering problem and other problems for spoken language MT, such as lots of omissions, idioms etc. Through experiments, we show that this approach obtains improvements over the baseline on a Chinese-English spoken language translation task.

1 Introduction

The state-of-the-art statistical machine translation (SMT) model is the log-linear model (Och and Ney, 2002), which provides a framework to incorporate any useful knowledge for machine translation, such as translation model, language model etc.

In a SMT system, one important problem is the re-ordering between words and phrases, especially when the source language and target language are very different in word order, such as Chinese and English.

For the spoken language translation, the re-ordering problem will be more crucial, since the spoken language is more flexible in word order. In addition, lots of omissions and idioms make the translation more difficult.

However, there exists some "useful" features, such as, most of the spoken text is shorter than the written text and there are some fixed translation structures. For example, (你能…? / Would you please … ?), (能…?/May I…?).

We can learn these fixed structures and take them as rules, Chiang (2005) presents a method to learn these rules, and uses them in the SMT. Generally, the number of these rules will be very large. In this paper, we propose an example-based decoder in a SMT model, which will use the translation examples to keep the translation structure, i.e. constraint the reordering, and make the omitted words having the chance to be translated.

The rest of this paper is organized as follows: Since our decoder is based on the inversion transduction grammars (ITG) (Wu, 1997), we introduce the ITG in Section 2 and describe the derived SMT model. In Section 3, we design the example-based decoder. In Section 4, we test our model and compare it with the baseline system. Then, we conclude in Section 5 and Section 6.

2 The SMT model

ITG is a synchronous context-free grammar, which generates two output streams simultaneously. It consists of the following five types of rules:

$$A \xrightarrow{p} [AA] \mid \langle AA \rangle \mid c_i / e_j \mid c_i / \varepsilon \mid \varepsilon / e_j \qquad (1)$$

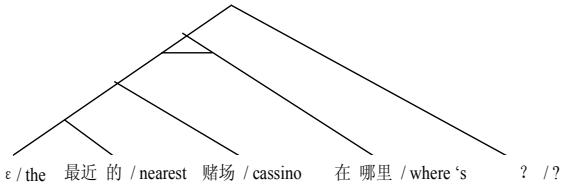
Where A is the non-terminal symbol, [] and <> represent the two operations which generate outputs in **straight** and **inverted** orientation respectively. c_i and e_j are terminal symbols, which represent the words in both languages, ε is the null

words. The last three rules are called lexical rules. p is the probability of the rule.

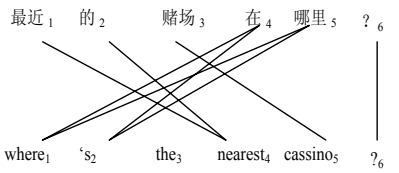
In this paper, we consider the phrase-based SMT, so the c_i and e_j represent phrases in both languages, which are consecutive words. And a pair of c_i and e_j is called a phrase-pair, or a **block**.

During the process of decoding, each phrase c_i in the source sentence is translated into a target phrase e_j through lexical rules, and then rules [] or <> are used to merge two adjacent blocks into a larger block in straight or inverted orientation, until the whole source sentence is covered. In this way, we will obtain a binary branching tree, which is different from the traditional syntactical tree, since each constituent in the branching tree is not a syntactical constituent.

Thus, the model achieves a great flexibility to interpret almost arbitrary reordering during the decoding, while keeping a weak but effective constraint. Figure 1(a) gives an example to illustrate a derivation from the ITG model.



(a) An ITG tree



(b) A word alignment

Figure 1. (a) An ITG tree derived from the ITG where the line between the branches means an inverted orientation, otherwise a straight one, (b) A word alignment corresponds to the ITG tree in (a).

Since we regard the process of the decoding as a sequence of applications of rules in (1), i.e., the output sentence pair (*C*,*E*) will be a derivation *D* of the ITG, where *C* represents the source sentence and *E* is the target sentence.

Following Och and Ney (2002), we define the probability for each rule as:

$$\Pr(rule) = \prod_i h_i(rule)^{\lambda_i} \quad (2)$$

Where the h_i represents the feature and λ_i is the corresponding weight of the feature.

We will consider mainly the following features for rules:

- Translation Models: $P(e|c)$, $P(c|e)$, $P_{lex}(e|c)$ and $P_{lex}(c|e)$. The first two models consider the probability of phrase translation; and the latter two consider the lexical translation, i.e., the probability that the words in source (or target) phrase translate to the ones in the target (or source) phrase.
- Reordering model: $P(o|b_1,b_2)$, where o is the output orientation and b_1, b_2 are the two blocks in the rule.
- Language model: $\Delta \Pr_{lm}(e)$, which considers the increment of the language model for each rule.

And the probability for the derivation will be:

$$\Pr(D) = \prod_{r \in D} \Pr(r) \quad (3)$$

So the decoder searches the best E^* derived from the best derivation D^*, when given a source sentence C.

$$D^* = \arg\max_{c(D)=C} \Pr(D) \quad (4)$$

2.1 Building the models

In our SMT model, we use the translation models and reordering model. They will be built from the training corpus, which is a word-aligned bilingual corpus satisfying the ITG constraint.

We define the word alignment A for the sentence pair (*C*,*E*) in the following ways:

- A region $(i..j, s..t)$: $i..j$ represents a sequence of position index in sentence *C*, i.e. $i, i+1, ..., j$ and $s..t$ represents a sequence of position index in sentence *E*, i.e. $s, s+1, ..., t$. We also call the $i..j$ and $s..t$ are regions in monolingual sentences. The region corresponds to a phrase pair, which we called as a block. The length of the block is $\max(|j-i+1|, |t-s+1|)$.

- A link $l=(i..j,s..t)$: And each link represents the alignment between the consecutive words in both of the sentences, which position indexes are in $i..j$ and $s..t$. If one of the $i..j$ and $s..t$ is ε, i.e. an empty region, we call the link a null-align.
- A word alignment *A*: a set of links $A=\{l_1,l_2,...,l_n\}$.

We can merge two links $l_1=(i_1..j_1,s_1..t_1)$ and $l_2=(i_2..j_2,s_2..t_2)$ to form a larger link, if the two links are adjacent in both of the sentences, i.e. $i_1..j_1$ is adjacent to $i_2..j_2$ where $i_2=j_1+1$ or $i_1=j_2+1$, or $i_1..j_1$ (or $i_2..j_2$) is ε , so do the $s_1..t_1$ to $s_2..t_2$. If the region $(i..j,s..t)$ can be formed by merging two adjacent links gradually, we call the region is independent, and the corresponding block is also independent.

In our system, the word alignment must satisfy the ITG constraint, i.e. the word alignment is able to form a binary branching tree. Figure 1(b) illustrates a word alignment example; the number below the word is the position index. In the example, the region (1..3, 3..5) is independent, and the block (*最近 的 赌场*，*the nearest cassino*) is also independent.

In order to obtain the word alignment satisfying the ITG constraint, Wu(1997) propose a DP algorithm, and we (Chao and Li, 2007) have transferred the constraint to four simple position judgment procedures in an explicit way, so that we can incorporate the ITG constraint as a feature into a log-linear word alignment model (Moore, 2005).

After obtaining the word-aligned corpus, in which each word alignment satisfy the ITG constraint, we can extract the blocks in a straightforward way. For the word alignment forms a hierarchical binary tree, we choose each constituent as a block. Each block is formed by combining one or more links, and must be independent. Considering the data sparseness, we limit the length of each block as N (here N=3~5).

We can also collect the reordering information between two blocks according to the orientation of the branches.

Thus, we will build the translation models $P(e|c)$, $P(c|e)$, $P_{lex}(e|c)$ and $P_{lex}(c|e)$, using the frequencies of the blocks, and the re-ordering model $P(o|b_1,b_2)$, $o\in\{straight,invert\}$ in the following way:

$$p(o|b_1,b_2)=\frac{\text{freq. of }(O(b_1,b_2)=o)}{\text{freq. of }cooccur\,(b_1,b_2)} \quad (5)$$

Considering the data sparseness, we transfer the re-ordering model in the following way:

$$p(o|b_1,b_2)=p(o|b_1,*)\bullet p(o|*,b_2) \quad (6)$$

where * represents any block, $p(o|b_1,*)$ represents the probability when $O(b_1,*)=o$, i.e., when b_1 occurs, the orientation it merges with any other block is o . So we can estimate the merging orientation through the two blocks respectively.

2.2 A Baseline Decoder

In order to evaluate the example-based decoder, we develop a CKY style decoder as a baseline (Chao et al. 2007), which will generate a derivation from the ITG in a DP way. And it is similar with the topical phrase-based SMT system, while maintaining the ITG constraint.

3 The Example-based Decoder

The SMT obtains the translation models during training, and does not need the training corpus when decoding; while the example-based machine translation system (EBMT) using the similar examples in the training corpus when decoding.

However, both of them use the same corpus; we can generate a hybrid MT, which is a SMT system while using an example-based decoder, to benefit from the advantages within the two systems.

Our example-based decoder consists of two components: retrieval of examples and decoding. Figure 2 shows the structure of the decoder.

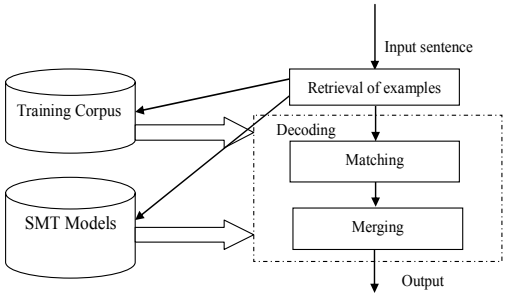


Figure 2. The structure of the example-based decoder.

3.1 Retrieval of Examples

Our training corpus is a sentence-aligned bilingual corpus. For each sentence pair (*C*,*E*), we obtained the word alignment *A*, satisfying the ITG constaint through the methods described in section 2. We call the triple (*C*,*A*,*E*) as an example.

So, the problem of retrieval of examples is: given the input source sentence C_0 and the training corpus, collecting a set of translation examples {(C_1, A_1, E_1) , (C_2, TA_2, E_2),....} from the corpus, where each translation example (C_i, A_i, E_i) is similar to the input sentence C_0.

The quality of the retrieval of the similar examples is very import to the hybrid MT. For the translating may run in a large-scale corpus and in a real-time way, we divide the retrieval of similar examples into two phases:

- Fast Retrieval Phase: retrieving the similar examples from the corpus quickly, and take them as candidates. The complexity should not be too high.
- Refining Phase: refining the candidates to find the most similar examples.

3.1.1 The Similarity Metric for Fast Retrieval

Given an input sentence $I = w_1 w_2 ... w_n$ and an example (*C*, *A*, *E*), we calculate the number of the matched source words between the input sentence and the source sentence C in the example firstly.

$$Sim_w(I, Exam) = \frac{2 * Match_w}{Len(I) + Len(C, A, E)} \quad (7)$$

where $Match_w$ is the number of the matched words and $Len(I)$ is the number of words in I, and $Len(C, A, E)$ is the number of the words in the in C.

Given an input sentence $I = w_1 w_2 ... w_n$, we obtain the relative blocks in the translation model for each word $w_i (i \in \{1,2,...n\}$. We use B^i_{k-gram} to represent the blocks, in which for each block (c, e), the source phrase c use the word w_i as the first word, and the length of c is k, i.e. the $c = w_{i..(i+k-1)}$. For each c, there may exists more than one blocks with c as the source phrase, so we will sort them by the probability and keep the best N (here set N=5) blocks. Now we represent the input sentence as:

$$\sigma(I) = \{b \mid b \in B^i_{k-gram}, 1 \le i \le n, 1 \le k \le n\} \quad (8)$$

For example, in an input sentence "我 来自 中国", $B^1_{1-gram} = \{(我, i), (我, me), (我, my), (我, Mine)\}$

Note, some B^i_{k-gram} may be empty, e.g. $B^2_{2-gram} = \phi$, since no blocks with "来自 中国" as the source phrase.

In the same way, we represent the example (C, A, E) as:

$$\varphi(C, A, E) = \{b \mid b \in B^i_{k-gram}, b \in A^*\} \quad (9)$$

where A^* represents the blocks which are links in the alignment A or can be formed by merging adjacent links independently. In order to accelerate the retrieval of similar examples, we generate the block set for the example during the training process and store them in the corpus.

Now, we can use the number of the matched blocks to measure the similarity of the input and the example:

$$Sim_b(I, Exam) = \frac{2 * Match_b}{B^I_{gram} + B^{Exam}_{gram}} \quad (10)$$

where $Match_b$ is the number of the matched blocks and B^I_{gram} is the number of B^i_{k-gram} ($B^i_{k-gram} \neq \phi$) in $\sigma(I)$, and B^{Exam}_{gram} is the number of the blocks in $\varphi(C, A, E)$.

Since each block is attached a probability, we can compute the similarity in the following way:

$$Sim_p(I, Exam) = \frac{2 * \sum_{b \in Match_b} \Pr ob(b)}{B^I_{gram} + B^{Exam}_{gram}} \quad (11)$$

So the final similarity metric for fast retrieval of the candidates is:

$$Sim_{fast}(I, Exam) = \alpha Sim_w + \beta Sim_b + \gamma Sim_p \quad (12)$$

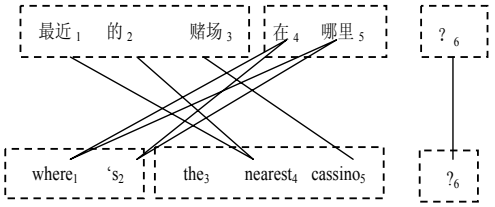
where $0 \le \alpha, \beta, \gamma \le 1$ $\alpha + \beta + \gamma = 1$. Here we use mean values, i.e. $\alpha = \beta = \gamma = 1/3$. During the fast retrieval phase, we first filter out the examples using the Sim_w, then calculate the Sim_{fast} for each example left, and retrieve the best *N* examples.

3.1.2 The Alignment Structure Metric

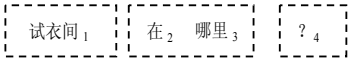
After retrieving the candidate similar examples, we refine the candidates using the word alignment structure with the example, to find the best *M* similar examples (here set *M*=10). The word alignment in the example satisfies the ITG constraint, which provides a weak structure constraint.

Given the input sentence I and an example (C, A, E), we first search the matched blocks, at this moment the order of the source phrases in the blocks must correspond with the order of the words in the input.

As Figure 3 shows, the matching divides the input and the example respectively into several regions, where some regions are matched and some un-matched. And we take each region as a whole and align them between the input and the example according to the order of the matched regions. For example, the region (1..3,3..5) in (C, A, E) is un-matched, which aligns to the region (1..1) in I. In this way, we can use a similar edit distance method to measure the similarity. We count the number of the Deletion / Insertion / Substitution operations, which take the region as the object.



(a) An example



(b) An input

Figure 3. An input and an example. After matching, there are three regions in both sides, which are included in the line box, where the region (4..5,1..2) in the example matches the region (2..3) in the input, so do (6..6,6..6) to (4..4). And the region (1..3,3..5) in the example should be substituted to (1..1) in the input.

We set the penalty for each deletion and insertion operation as 1, while considering the un-matched region in the example may be independent or not, we set the penalty for substitution as 0.5 if the region is independent, otherwise as 1. E.g., the distance is 0.5 for substituting the region (1..3,3..5) to (1..1).

We get the metric for measuring the structure similarity of the I and (C, A, E):

$$Sim_{align}(I, Exam) = 1 - \frac{D + I + S}{R_{input} + R_{exmaple}} \qquad (13)$$

where D, I, S are the deletion, insertion and substitution distances, respectively. And the R_{input} and $R_{exmaple}$ are the region numbers in the input and example.

In the end, we obtain the similarity metric, which considers all of the above metrics:

$$Sim_{final}(I, Exam) = \alpha' Sim_{fast} + \beta' Sim_{align} \qquad (14)$$

where $0 \le \alpha', \beta' \le 1$ $\alpha' + \beta' = 1$. Here we also use mean values $\alpha' = \beta' = 1/2$.

After the two phrases, we obtain the most similar examples with the input sentence.

3.2 Decoding

After retrieving the translation examples, our goal is to use these examples to constrain the order of the output words. During the decoding, we iterate the following two steps.

3.2.1 Matching

For each translation example (C_k, A_k, E_k) consists of the constituent structure tree, we can match the input sentence with the tree as in Section 3.1.2.

After matching, we obtain a translation of the input sentence, in which some input phrases are matched to blocks in the tree, i.e. they are translated, and some phrases are un-translated. The order of the matched blocks must be the same as the input phrases. We call the translation as a translation template for the input.

If we take each un-translated phrase as a null-aligned block, the translation template will be able to form a new constituent tree. And the matched blocks in the template will restrict the translation structure.

Figure 4(a-c) illustrates the matching process, and Figure 4(c) is a translation template, in which "你 能" and "吗 ？ " have been translated and "打开 你的 包" is not translated. And the translation

template can be derived from the ITG as follows (here we remove the un-matched phrase):

$$
\begin{aligned}
&A -> [A_1 A_2] \\
&A_1 -> \langle A_3 A_4 \rangle \\
&A_2 -> 吗?/? \\
&A_3 -> 你 / you \\
&A_4 -> 能 / could
\end{aligned}
\tag{15}
$$

Since we have M (here M=10) similar examples, we will get more than one translation template for the input sentence. So we define the evaluation function f for each translation template as :

$$f(temp) = \log P(D_{trans}) + \log H(C_{untrans}) \tag{16}$$

Where $P(D_{trans})$ is the probability for the new ITG tree without the un-translated phrases, which is a derivation from the ITG, so we can calculate it using the SMT model in Section 2 (formula 3).

And the $H(C_{untrans})$ is the estimated score for the un-translated phrases. In order to obtain $H(C_{untrans})$, we estimate the score for each un-translated phrase $c_{m..n}$ in the following way:

$$H(c_{m..n}) = \max\{\max_{k} H(c_{m..k}) \bullet H(c_{k..n}), \max_{e^*} p(e^* \mid c_{m..n})\} \tag{17}$$

That is, using the best translation to estimate the translation score. Thus we can estimate the $H(C_{untrans})$ as:

$$H(C_{untrans}) = \prod_{c} H(c_{m..n}) \tag{18}$$

We call the un-translated phrases as child inputs, and try to translate them literately, i.e., decoding them using the examples. If there are no un-translated phrases in the input, the decoding is completed, and the decoder returns the translation template with the best score as the result.

3.2.2 Merging

If one child input is translated completely, i.e. no phrase is un-translated. Then, it should be merged into the parent translation template to form a new template. When merging, we must satisfy the ITG constraint, so we use the rules [] and <> to merge the child input with the adjacent blocks. Figure 4(c-f) illustrates a merging process.

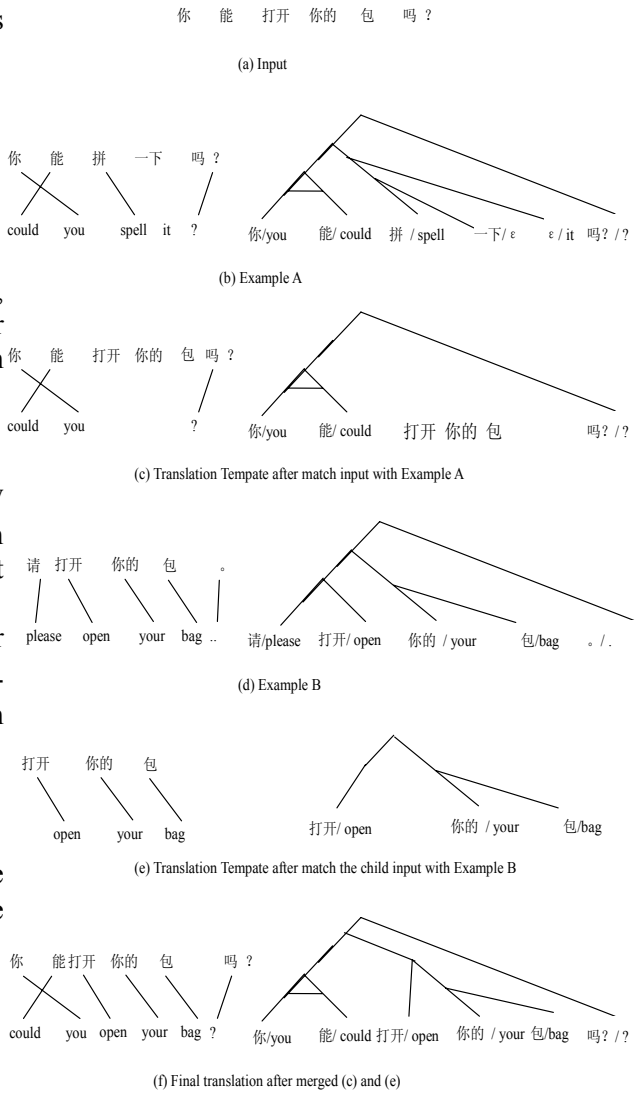


Figure 4. An example to illustrate the example-based decoding process, in which there are two translation examples.

When merging, it may modify some rules which are adjacent to the child inputs. For example, when merging Figure 4(c) and (e), we may add a new rule:

$$A_1' -> [A_1 A_{child}] \tag{19}$$

A_{child} is the root non-terminal for the child input. And we should modify the rule $A -> [A_1 A_2]$ as:

$$A -> [A_1' A_2] \tag{20}$$

The merged template may vary due to the following situations:

- The orientation may vary. The orientation between the new block formed from the child

template and the preceding or posterior blocks may be straight or inverted.

- The position to merge may vary. We may merge the new block with either the whole preceding or posterior blocks, or only the child blocks of them respectively, i.e. we may take the preceding or posterior blocks as the whole blocks or not.

Thus, we will obtain a collection of the merged translation templates, the decoder will evaluate them using the formualte (16). If all the templates have no un-translated phrases, return the template with the best score.

3.2.3 Decoding Algorithm

The decoding algorithm is showed in Figure 5.

In line 5~8, we match the input sentence with each similar example, and generate a collection of translation templates, using the formular (16) to evaluate the templates.

In line 9~11, we verify whether the set of the templates for the input is null: If it is null, decoding the input using the normal CKY decoder, and return the translations.

In lin 12~23, we decode the un-matched phrase in each template, and merge it with the parent template, until all of the template are translated completely.

In line 24, we return the best N translations.

4 Experiments

We carried out experiments on an open Chinese-English translation task IWSLT2007, which consisting of sentence-aligned spoken language text for traveling. There are five development set, and we take the third development set, i.e. the *IWSLT07_devset3_**, to tune the feature weights.

		Chinese	English stemmed
Train. corpus	Sentences	39,963	
	Words	351,060	377,890
	Vocabulary	11,302	7,610
Dev. Set	Sentences	506	
	Words	3,826	
Test Set	Sentences	489	
	Words	3,189	

Table 1. The statistics of the corpus

```
1: Function Example_Decoder(I,examples)
2: Input: Input sentence I, Similar Examples examples
3: Output: The best N tranlsations
4: Begin
5:  For each exampleA in examples Do
6:    templates = Match(exampleA,I);
7:    AddTemplate(templates,I);
8:  End {For}
9:  If templates is null then
10:   templates = CYK_Decoder(I);
11:   return templates;
12: For each templateA in templates Do
13:  If templateA is complete then
14:    AddTemplate_Complete(templateA,I);
15:  Else
16:    RemoveTemplate(templateA,I);
17:    For each untranslated phraseB in templateA do
18:      childTemplates = Example_Decoder(phraseB);
19:      For each childTemplateC in childTemplates Do
20:        templateD=MergeTemplate(templateA,childTemplateC);
21:   End{If}
22:   AddTemplate(templateD,I);
23:  End{For}
24:  return BEST_N(complete_templates);
28: End
```

Figure 5. The decoding algorithm.

Considering the size of the training corpus is relatively small, and the words in Chinese have no morphological changes, we stemmed the words in the English sentences.

Table 1 shows the statistics for the training corpus, development set and test set.

In order to compare with the other SMT systems, we choose the Moses[1], which is an extension to the state-of-the-art SMT system Pharaoh (Koehn, 2004). We use the default tool in the Moses to train the model and tune the weights, in which the word alignment tool is Giza++ (Och and Ney 2003) and the language model tool is SRILM(Stolcke, 2002).

The test results are showed in Table2.

The first column lists the different MT systems, and the second column lists the Bleu scores (Papineni et. al, 2002) for the four decoders.

The first system is the Moses, and the second is our SMT system described in section 2, which using a CKY-style decoder. We take them as baseline systems. The third is the hybrid system but

[1] http://www.statmt.org/moses/.

only using the fast retrieval module and the fourth is the hybrid system with refined retrieval module.

Considering the result from the Moses, we think that maybe the size of the training corpus is too small, so that the word alignment obtained by Giza++ is poor.

The results show that the example-based decoder achieves an improvement over the baseline decoders.

Decoder	Bleu
Moses	22.61
SMT-CKY	28.33
Hybrid MT with fast retrieval	30.03
Hybrid MT with refined retrieval	33.05

Table 2. Test results for several systems.

5 Related works

There is some works about the hybrid machine translation. One way is to merge EBMT and SMT resources, such as Groves and Way (2005).

Another way is to implement an exmaple-based decoder, Watanabe and Sumita (2003) presents an example-based decoder, which using a information retrieval framework to retrieve the examples; and when decoding, which runs a hill-climbing algorithm to modify the translation example (C_k, E_k, A_k) to obtain an alignment (C_0, E'_k, A'_k).

6 Conclusions

In this paper, we proposed a SMT system with an example-based decoder for the spoken language machine translation. This approach will take advantage of the constituent tree within the translation examples to constrain the flexible word reordering in the spoken language, and it will also make the omitted words have the chance to be translated. Combining with the re-ordering model and the translation models in the SMT, the example-based decoder obtains an improvement over the baseline phrase-based SMT system.

In the future, we will test our method in the written text corpus. In addition, we will improve the methods to handle the morphological changes from the stemmed English words.

Acknowledgements

This work is supported by the National Science Foundation of China under Grants No. 60573057, 60473057 and 90604007.

References

Wen-Han Chao and Zhou-Jun Li.(2007). *Incorporating Constituent Structure Constraint into Discriminative Word Alignment.* MT Summit XI, Copenhagen, Denmark, September 10-14, 2007. pp.97-103.

Wen-Han Chao, Zhou-Jun Li, and Yue-Xin Chen.(2007) *An Integrated Reordering Model for Statistical Machine Translation.* In proceedings of MICAI 2007, LNAI 4827, pp. 955–965, 2007.

David Chiang. (2005). *A Hierarchical Phrase-Based Model for Statistical Machine Translation.* In Proc. of ACL 2005, pages 263–270.

Declan Groves and Andy Way: *Hybrid Example-Based SMT: the Best of Both Worlds?* In Proceedings of the ACL Workshop on Building and Using Parallel Texts, pp. 183-190(2005)

P. Koehn.(2004) Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In: Proceedings of the Sixth Conference of the Association for Machine Translation in the Americas, pp. 115–124.

R. Moore. (2005). *A discriminative framework for bilingual word alignment.* In Proceedings of HLT-EMNLP, pages 81–88, Vancouver, Canada, October.

Franz Joseph Och and Hermann Ney.(2002). *Discriminative training and maximum entropy models for statistical machine translation.* In Proceedings of the 40th Annual Meeting of the ACL, pp. 295–302.

Franz Joseph Och and Hermann Ney. (2003) *A Systematic Comparison of Various Statistical Alignment Models.* Computational Linguistics 29(1), 19–52

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. (2002). *BLEU: a Method for Automatic Evaluation of Machine Translation.* In Proceedings of the 40th Annual Meeting of the Association fo Computational Linguistics (ACL), Philadelphia, July 2002, pp. 311-318.

A. Stolcke. (2002). *SRILM – An extensible language modeling toolkit.* In Proceedings of the International Conference on Spoken Language Processing, Denver, Colorado, 2002, pp. 901–904.

Taro Watanabe and Eiichiro Sumita. (2003). *Example-based Decoding for Statistical Machine Translation.* In Machine Translation Summit IX pp. 410-417.

Dekai Wu. (1997). *Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora.* Computational Linguistics, 23(3):374.