Advancements in Reordering Models for Statistical Machine Translation

Minwei Feng and Jan-Thorsten Peter and Hermann Ney
Human Language Technology and Pattern Recognition
Computer Science Department, RWTH Aachen University, Aachen, Germany
<surname>@cs.rwth-aachen.de

Abstract
In this paper, we propose a novel reordering model based on sequence labeling techniques. Our model converts the reordering problem into a sequence labeling problem, i.e., a tagging task. Results on five Chinese-English NIST tasks show that our model improves the baseline system by 1.32 BLEU and 1.53 TER on average. Results of comparative study with other seven widely used reordering models will also be reported.

1 Introduction
The systematic word order difference between two languages poses a challenge for current statistical machine translation (SMT) systems. The system has to decide in which order to translate the given source words. This problem is known as the reordering problem. As shown in (Knight, 1999), if arbitrary reordering is allowed, the search problem is NP-hard.

Many ideas have been proposed to address the reordering problem. Within the phrase-based SMT framework there are mainly three stages where improved reordering could be integrated:

In the preprocessing: the source sentence is reordered by heuristics, so that the word order of source and target sentences is similar. (Wang et al., 2007) use manually designed rules to reorder parse trees of the source sentences. Based on shallow syntax, (Zhang et al., 2007) use rules to reorder the source sentences on the chunk level and provide a source-reordering lattice instead of a single reordered source sentence as input to the SMT system. Designing rules to reorder the source sentence is conceptually clear and usually easy to implement. In this way, syntax information can be incorporated into phrase-based SMT systems. However, one disadvantage is that the reliability of the rules is often language pair dependent.

In the decoder: we can add constraints or models into the decoder to reward good reordering options or penalize bad ones. For reordering constraints, early work includes ITG constraints (Wu, 1995) and IBM constraints (Berger et al., 1996). (Zens and Ney, 2003) did comparative study over different reordering constraints. This paper focuses on reordering models. For reordering models, we can further roughly divide the existing methods into three genres:

• The reordering is a classification problem. The classifier will make decision on next phrase’s relative position with current phrase. The classifier can be trained with maximum likelihood like Moses lexicalized reordering (Koehn et al., 2007) and hierarchical lexicalized reordering model (Galley and Manning, 2008) or be trained under maximum entropy framework (Zens and Ney, 2006).

• The reordering is a decoding order problem. (Mariño et al., 2006) present a translation model that constitutes a language model of a sort of bilanguage composed of bilingual units. From the reordering point of view, the idea is that the correct reordering is a suitable order of translation units. (Feng et al., 2010) present a simpler version of (Mariño et al., 2006)’s model which utilize only source words to model the decoding order.

• The reordering can be solved by outside heuristics. We can put human knowledge into the decoder. For example, the simple jump model using linear distance tells the decoder that usually the long range reordering should be avoided. (Cherry, 2008) uses information from dependency trees to make the decoding process keep syntactic cohesion. (Feng et al., 2012) present a method that utilizes predicate-argument structures from semantic role labeling results as soft constraints.

In the reranking framework: in principle, all
the models in previous category can be used in the reranking framework, because in the reranking we have all the information (source and target words/phrases, alignment) about the translation process. (Och et al., 2004) describe the use of syntactic features in the rescoring step. However, they report the syntactic features contribute very small gains. One disadvantage of carrying out reordering in reranking is the representativeness of the N-best list is often a question mark.

In this paper, we propose a novel tagging style reordering model which is under the category “The reordering is a decoding order problem”. Our model converts the decoding order problem into a sequence labeling problem, i.e. a tagging task. The remainder of this paper is organized as follows: Section 2 introduces the basement of this research: the principle of statistical machine translation. Section 3 describes the proposed model. Section 4 briefly describes several reordering models with which we compare our method. Section 5 provides the experimental configuration and results. Conclusion will be given in Section 6.

2 Translation System Overview

In statistical machine translation, we are given a source language sentence \( f _ 1 ^ J = f _ 1 \ldots f _ j \ldots f _ J \). The objective is to translate the source into a target language sentence \( e _ 1 ^ J = e _ 1 \ldots e _ i \ldots e _ J \). The strategy is to choose the target sentence with the highest probability among all others:

\[
\hat{e} _ i ^ J = \arg \max _ { f _ i ^ J } \{ \Pr ( e _ i ^ J | f _ i ^ J ) \}
\]

We model \( \Pr ( e _ i ^ J | f _ i ^ J ) \) directly using a log-linear combination of several models (Och and Ney, 2002):

\[
\Pr ( e _ i ^ J | f _ i ^ J ) = \frac{\exp \left( \sum _ { m = 1 } ^ { M } \lambda _ m h _ m ( e _ i ^ J , f _ i ^ J ) \right)}{\sum _ { i ^ \prime , e _ i ^ { \prime J } } \exp \left( \sum _ { m = 1 } ^ { M } \lambda _ m h _ m ( e _ i ^ { \prime J } , f _ i ^ { \prime J } ) \right)}
\]

The denominator is to make the \( \Pr ( e _ i ^ J | f _ i ^ J ) \) to be a probability distribution and it depends only on the source sentence \( f _ i ^ J \). For search, the decision rule is simply:

\[
\hat{e} _ i ^ J = \arg \max _ { f _ i ^ J } \left\{ \sum _ { m = 1 } ^ { M } \lambda _ m h _ m ( e _ i ^ J , f _ i ^ J ) \right\}
\]

The model scaling factors \( \lambda _ m ^ J \) are trained with Minimum Error Rate Training (MERT). In this paper, the phrase-based machine translation system is utilized (Och et al., 1999; Zens et al., 2002; Koehn et al., 2003).

3 Tagging-style Reordering Model

In this section, we describe the proposed novel model. First we will describe the training process. Then we explain how to use the model in the decoder.

3.1 Modeling

Figure 1 shows the modeling steps. The first step is word alignment training. Figure 1(a) is an example after GIZA++ training. If we regard this alignment as a translation result, i.e. given the source sentence \( f _ 1 ^ J \), the system translates it into the target sentence \( e _ 1 ^ J \), then the alignment link set \( \{ a _ 1 = 3, a _ 3 = 2, a _ 4 = 4, a _ 5 = 7, a _ 6 = 6, a _ 7 = 6 \} \) reveals the decoding process, i.e. the alignment implies the order in which the source words should be translated, e.g. the first generated target word \( e _ 1 \) has no alignment, we can regard it as a translation from a NULL source word; then the second generated target word \( e _ 2 \) is translated from \( f _ 3 \). We reorder the source side of the alignment to get Figure 1(b). Figure 1(b) implies the source sentence decoding sequence information, which is depicted in Figure 1(c). Using this example we describe the strategies we used for special cases in the transformation from Figure 1(b) to Figure 1(c):

- ignore the unaligned target word, e.g. \( e _ 1 \)
- the unaligned source word should follow its preceding word, the unaligned feature is kept with a * symbol, e.g. \( f _ 2 ^ * \) is after \( f _ 1 \)
- when one source word is aligned to multiple target words, only keep the alignment that links the source word to the first target word, e.g. \( f _ 4 \) is linked to \( e _ 5 \) and \( e _ 6 \), only \( f _ 4 - e _ 5 \) is kept. In other words, we use this strategy to guarantee that every source word appears only once in the source decoding sequence.
- when multiple source words are aligned to one target word, put together the source words according to their original relative positions, e.g. \( e _ 6 \) is linked to \( f _ 6 \) and \( f _ 7 \). So in the decoding sequence, \( f _ 6 \) is before \( f _ 7 \).

Now Figure 1(c) shows the original source sentence and its decoding sequence. By using the strategies above, it is guaranteed that the source sentence and its decoding sequence have the ex-
The long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) is applied to sequence from both directions before making the prediction. The long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) is applied to sequence from both directions before making the prediction. The long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) is applied to sequence from both directions before making the prediction. The long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) is applied to sequence from both directions before making the prediction.

For the first method, we adopt the linear-chain CRFs. However, even for the simple linear-chain CRFs, the complexity of learning and inference grows quadratically with respect to the number of output labels and the amount of structural features which are with regard to adjacent pairs of labels. Hence, to make the computational cost as low as possible, two measures have been taken. Firstly, as described above we reduce the number of tags to nine. Secondly, we add source sentence part-of-speech (POS) tags to the input. For features with window size one to three, both source words and its POS tags are used. For features with window size four and five, only POS tags are used.

As the second method, we use recurrent neural network (RNN). RNN is closely related with Multilayer Perceptrons (MLP) (Rumelhart et al., 1986), but the output of one ore more hidden layers is reused as additional inputs for the network in the next time step. This structure allows the RNN to learn whole sequences without restricting itself to a fixed input window. A plain RNN has only access to the previous events in the input sequence. Hence we adopt the bidirectional RNN (BRNN) (Schuster and Paliwal, 1997) which reads the input sequence from both directions before making the prediction. The long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) is applied to
counter the effects that long distance dependencies are hard to learn with gradient descent. This is often referred to as vanishing gradient problem (Bengio et al., 1994).

3.2 Decoding

Once the model training is finished, we make inference on develop and test corpora which means that we get the labels of the source sentences that need to be translated. In the decoder, we add a new model which checks the labeling consistency when scoring an extended state. During the search, a sentence pair \( (f^I_1, e^I_1) \) will be formally split into a segmentation \( S^I_1^K \) which consists of \( K \) phrase pairs. Each \( s_k = (i_k; b_k, j_k) \) is a triple consisting of the last position \( i_k \) of the \( k \)th target phrase \( e_k \). The start and end position of the \( k \)th source phrase \( f_k \) are \( b_k \) and \( j_k \).

Suppose the search state is now extended with a new phrase pair \( (f_k, e_k) \): \( f_k := f_{b_k} \ldots f_{j_k} \) and \( e_k := e_{i_{k-1}+1} \ldots e_{i_k} \). We have access to the old coverage vector, from which we know if the new phrase’s left neighboring source word \( f_{b_k-1} \) and right neighboring source word \( f_{j_k+1} \) have been translated. We also have the word alignment within the new phrase pair, which is stored during the phrase extraction process. Based on the old coverage vector and alignment, we can repeat the transformation in Figure 1 to calculate the labels for the new phrase. The added model will then check the consistency between the calculated labels and the labels predicted by the reordering model. The number of source words that have inconsistent labels is the penalty and is then added into the log-linear framework as a new feature.

4 Comparative Study

The second part of this paper is comparative study on reordering models. Here we briefly describe those models which will be compared to later.

4.1 Moses lexicalized reordering model

Moses (Koehn et al., 2007) contains a word-based orientation model, which has three types of reordering: (m) monotone order, (s) switch with previous phrase and (d) discontinuous. Figure 2 is an example. The definitions of reordering types are as follows:

- **monotone** for current phrase, if a word alignment to the bottom left (point A) exists and there is no word alignment point at the bottom right position (point B).
- **swap** for current phrase, if a word alignment to the bottom right (point B) exists and there is no word alignment point at the bottom left position (point A).
- **discontinuous** all other cases

Our implementation is same with the default behavior of Moses lexicalized reordering model. We count how often each extracted phrase pair is found with each of the three reordering types. The add-0.5 smoothing is then applied. Finally, the probability is estimated with maximum likelihood principle.

4.2 Maximum entropy reordering model

Figure 3 is an illustration of (Zens and Ney, 2006). \( j \) is the source word position which is aligned to the last target word of the current phrase. \( j' \) is the last source word position of the current phrase. \( j'' \) is the source word position which is aligned to the first target word position of the next phrase. (Zens and Ney, 2006) proposed a maximum entropy classifier to predict the orientation of the next phrase given the current phrase. The orientation class \( c_{j,j',j''} \) is defined as:

\[
c_{j,j',j''} = \begin{cases} 
  \text{left,} & \text{if } j' < j \\
  \text{right,} & \text{if } j'' > j \text{ and } j'' - j' > 1 \\
  \text{monotone,} & \text{if } j'' > j \text{ and } j'' - j' = 1
\end{cases}
\]

The orientation probability is modeled in a log-linear framework using a set of \( N \) feature functions \( h_n(f^I_1, e^I_1, i, j, c_{j,j',j''}) \), \( n = 1, \ldots, N \). The whole model is:

\[
p_{\lambda N}^N(c_{j,j',j''} | f^I_1, e^I_1, i, j) = \exp\left(\sum_{n=1}^N \lambda_n h_n(f^I_1, e^I_1, i, j, c_{j,j',j''})\right) \tag{5}
\]

Different features can be used, we use the source and target word features to train the model.
Figure 3: phrase orientation: left, right and monotone. $j$ is the source word position aligned to the last target word of current phrase. $j$ is the last source word position of current phrase. $j''$ is the source word position aligned to the first target word position of the next phrase.

Figure 4: bilingual LM illustration. The bilingual sequence is $e_1, e_2, f_3, e_3, f_1, e_4, f_4, e_5, f_6, f_7, e_6, f_6, f_7, e_7, f_5$.

4.3 Bilingual LM

The previous two models belong to “The reordering is a classification problem”. Now we turn to “The reordering is a decoding order problem”. (Mariño et al., 2006) implement a translation model using $n$-grams. In this way, the translation system can take full advantage of the smoothing and consistency provided by standard back-off $n$-gram models. Figure 4 is an example. The interpretation is that given the sentence pair $(f_1^1, e_1^1)$ and its alignment, the correct translation order is $e_1, e_2, f_3, e_3, f_1, e_4, f_4, e_5, f_6, f_7, e_6, f_6, f_7, e_7, f_5$. Notice the bilingual units have been ordered according to the target side, as the decoder writes the translation in a left-to-right way. Using the example we describe the strategies used for special cases:

- keep the unaligned target word, e.g. $e_1$
- remove the unaligned source word, e.g. $f_2$
- when one source word aligned to multiple target words, duplicate the source word for each target word, e.g. $e_4, f_4, e_5, f_4$
- when multiple source words aligned to one target word, put together the source words for that target word, e.g. $e_6, f_6, f_7$

After the operation in Figure 4 was done for all bilingual sentence pairs, we get a decoding sequence. In search, the bilingual LM is applied similar to the standard target side LM. The bilingual sequence of phrase pairs will be extracted using the same strategy in Figure 4. Suppose the search state is now extended with a new phrase pair $(\tilde{f}, \tilde{e})$. $\tilde{F}$ is the bilingual sequence for the new phrase pair $(\tilde{f}, \tilde{e})$ and $\tilde{F}^i$ is the $i^{th}$ unit within $\tilde{F}$. $\tilde{F}'$ is the bilingual sequence history for current state. We compute the feature score $h_{bilm}(\tilde{F}, \tilde{F}')$ of the extended state as follows:

$$h_{bilm}(\tilde{F}, \tilde{F}') = \lambda \sum_{i=1}^{\lvert \tilde{F} \rvert} \log p(\tilde{F}^1, \tilde{F}^2, \cdots, \tilde{F}^{i-1})$$

(6)

$\lambda$ is the scaling factor for this model. $\lvert \tilde{F} \rvert$ is the length of the bilingual decoding sequence.

4.4 Source decoding sequence LM

(Feng et al., 2010) present an simpler version of the above bilingual LM where they use only the source side to model the decoding order. The source word decoding sequence in Figure 4 is then $f_3, f_1, f_2, f_4, f_6, f_7, f_5$. We also build a 9-gram LM based on the source word decoding sequences. The usage of the model is same as bilingual LM.

4.5 Syntactic cohesion model

The previous two models belong to “The reordering is a decoding order process”. Now we turn to “The reordering can be solved by outside heuristics”. (Cherry, 2008) proposed a syntactic cohesion model. The core idea is that the syntactic structure of the source sentence should be preserved during translation. This structure is represented by a source sentence dependency tree. The algorithm is as follows: given the source sentence and its dependency tree, during the translation process, once a hypothesis is extended, check if the source dependency tree contains a subtree $T$ such that:
• Its translation is already started (at least one node is covered)
• It is interrupted by the new added phrase (at least one word in the new source phrase is not in \( T \))
• It is not finished (after the new phrase is added, there is still at least one free node in \( T \))

If so, we say this hypothesis violates the subtree \( T \), and the model returns the number of subtrees that this hypothesis violates.

4.6 Semantic cohesion model

(Feng et al., 2012) propose two structure features from semantic role labeling (SRL) results. Similar to the previous model, the SRL information is used as soft constraints. During decoding process, the first feature will report how many event layers that one search state violates and the second feature will report the amount of semantic roles that one search state violates. In this paper, the two features have been used together. So when the semantic cohesion model is used, both features will be triggered.

4.7 Tree-based jump model

(Wang et al., 2007) present a pre-reordering method for Chinese-English translation task. In Section 3.6 of (Zhang, 2013), instead of doing hard reordering decision, the author uses the rules as soft constraints in the decoder. In this paper, we use the similar method as described in (Zhang, 2013). Our strategy is: firstly, we parse the source sentences to get constituency trees. Then we manipulate the trees using heuristics described by (Wang et al., 2007). The leaf nodes in the revised tree constitute the reordered source sentence. Finally, in the log-linear framework (Equation 2) a new jump model is added which uses the reordered source sentence to calculate the cost. For example, the original sentence \( f_1 f_2 f_3 f_4 f_5 \) is now converted by rules into the new sentence \( f_1 f_3 f_5 f_3 f_4 \). For decoding, we still use the original sentence. Suppose previously translated source phrase is \( f_1 \) and the current phrase is \( f_5 \). Then the standard jump model gives cost \( q_{Dist} = 4 \) and the new tree-based jump model will return a cost \( q_{Dist,new} = 1 \).

5 Experiments

In this section, we describe the baseline setup, the CRFs training results, the RNN training results and translation experimental results.

5.1 Experimental Setup

Our baseline is a phrase-based decoder, which includes the following models: an \( n \)-gram target-side language model (LM), a phrase translation model and a word-based lexicon model. The latter two models are used for both directions: \( p(f|e) \) and \( p(e|f) \). Additionally we use phrase count features, word and phrase penalty. The reordering model for the baseline system is the distance-based jump model which uses linear distance. This model does not have hard limit. We list the important information regarding the experimental setup below. All those conditions have been kept same in this work.

- lowercased training data from the GALE task (Table 1, UN corpus not included)
- alignment trained with GIZA++
- tuning corpus: NIST06 test corpora: NIST02 03 04 05 and 08
- 5-gram LM (1 694 412 027 running words) trained by SRILM toolkit (Stolcke, 2002) with modified Kneser-Ney smoothing training data: target side of bilingual data.
- BLEU (Papineni et al., 2001) and TER (Snover et al., 2005) reported all scores calculated in lowercase way.
- Wapiti toolkit (Lavergne et al., 2010) used for CRFs; RNN is built by the RNNLIB toolkit.

| Sentences | Chinese | English |
|-----------|---------|---------|
| Running Words | 115 172 748 | 129 820 318 |
| Vocabulary | 1 125 437 | 739 251 |

Table 1: translation model and LM training data statistics

Table 1 contains the data statistics used for translation model and LM. For the reordering model, we take two further filtering steps. Firstly, we delete the sentence pairs if the source sentence length is one. When the source sentence has only one word, the translation will be always monotonic and the reordering model does not need to learn this. Secondly, we delete the sentence pairs if the source sentence contains more than three contiguous unaligned words. When this happens, the sentence pair is usually low quality hence not suitable for learning. The main purpose of the two filtering steps is to further lay down the computational burden. The label distribution is depicted in Figure 5. We can see that most words are monotonic. We then divide the corpus to three parts:
5.2 CRFs Training Results

The toolkit Wapiti (Lavergne et al., 2010) is used in this paper. We choose the classical optimization algorithm limited memory BFGS (L-BFGS) (Liu and Nocedal, 1989). For regularization, Wapiti uses both the $\ell^1$ and $\ell^2$ penalty terms, yielding the elastic-net penalty of the form

$$
\rho_1 \parallel \theta \parallel_1 + \frac{\rho_2}{2} \parallel \theta \parallel_2^2
$$

In this work, we use as many features as possible because $\ell^1$ penalty $\rho_1 \parallel \theta \parallel_1$ is able to yield sparse parameter vectors, i.e. using a $\ell^1$ penalty term implicitly performs the feature selection. The computational costs are given here: on a cluster with two AMD Opteron(tm) Processor 6176 (total 24 cores), the training time is about 16 hours, peak memory is around 120G. Several experiments have been done to find the suitable hyperparameter $\rho_1$ and $\rho_2$, we choose the model with lowest error rate on validation corpus for translation experiments. The error rate of the chosen model on test corpus (the test corpus in Table 2) is 25.75% for token error rate and 69.39% for sequence error rate.

5.3 RNN Training Results

We also applied RNN to the task as an alternative approach to CRFs. The here used RNN implementation is RNRLIB which has support for long short term memory (LSTM) (Graves, 2008). We used a one of k encoding for the input word and also for the labels. After testing several configurations over the validation corpus we used a network with

| Feature Templates |
|--------------------|
| 1-gram source word features |
| $x[-4,0]/x[-3,0]/x[-2,0]/x[-1,0]/x[0,0]$ |
| $x[-4,0]/x[-3,0]/x[-2,0]/x[-1,0]/x[0,1]$ |
| 1-gram source POS features |
| $x[0,0]/x[1,0]/x[2,0]/x[3,0]/x[4,0]$ |
| 2-gram source word features |
| $x[-1,0]/x[0,0]/x[1,0]$ |
| $x[-2,0]/x[-1,0]/x[0,1]$ |
| 3-gram source word features |
| $x[-1,0]/x[0,0]/x[1,0]$ |
| $x[-2,0]/x[-1,0]/x[0,0]$ |
| 4-gram source word features |
| $x[-1,1]/x[0,1]/x[1,1]/x[2,1]$ |
| $x[-2,1]/x[-1,1]/x[0,2]/x[1,2]$ |
| 5-gram source word features |
| $x[-1,2]/x[-1,1]/x[0,3]/x[1,3]/x[2,3]$ |
| bingram output label feature |
| $x[-1,2]/x[0,2]$ |

Table 4: feature examples. $x[\text{row}, \text{col}]$ specifies a token in the input data. $\text{row}$ specifies the relative position from the current label and $\text{col}$ specifies the absolute position of the column. So for the current label in this table, $x[-1, 2]/x[0, 2]$ is Lmono-Rmono/UNALIGN and $x[-1, 1]/x[0, 1]/x[1, 1]$ is PN/VC/VV.

LSTM 200 nodes in the hidden layer. The RNN has a token error rate of 27.31% and a sentence error rate of 77.00% over the test corpus in Table 2. The RNN is trained on a similar computer as above. RNRLIB utilizes only one thread. The training time is about three and a half days and peak memory consumption is 1G.

5.4 Comparison of CRFs and RNN errors

CRFs performs better than RNN (token error rate 25.75% vs 27.31%). Both error rate values are much higher than what we usually see in part-of-speech tagging task. The main reason is that the “annotated” corpus is converted from word alignment which contains lots of error. However, as we
Table 5: CRF Confusion Matrix. Abbreviations: Lmono(Lm) Lreorder(Lr) Rmono(Rm) Rreorder(Rr)

| Reference | Prediction | Unalign | BEGIN-Rm | BEGIN-Rr | END-Lm | END-Lr | Lm-Rm | Lr-Rm | Lm-Rr | Lr-Rr |
|-----------|------------|---------|----------|----------|--------|--------|-------|-------|-------|-------|
| Unalign   | 687724     | 15084   | 850      | 7347     | 716    | 493984 | 107364| 43457 | 9194  |
| BEGIN-Rmono | 45137     | 338315  | 6209     | 0        | 0      | 0      | 0     | 0     | 0     |
| BEGIN-Rreorder | 419      | 12557   | 17054    | 0        | 0      | 0      | 0     | 0     | 0     |
| END-Lmono | 1799       | 0       | 0        | 365635   | 3196   | 0      | 0     | 0     | 0     |
| END-Lreorder | 510       | 0       | 0        | 5239     | 7911   | 0      | 0     | 0     | 0     |
| Rmono-Rmono | 188627    | 0       | 0        | 0        | 0      | 403278 | 176852| 150952| 13114 |
| Lmono-Rreorder | 88177    | 0       | 0        | 0        | 0      | 369212 | 430272| 27162 | 15275 |
| Rmono-Rreorder | 32342    | 0       | 0        | 0        | 0      | 266570 | 24558 | 296033| 10645 |
| Lreorder-Rreorder | 9865    | 0       | 0        | 0        | 0      | 34746  | 20382 | 16514 | 45347 |

Recall: 50.36% 97.20% 56.79% 98.65% 57.92% 88.40% 46.42% 46.83% 35.74%
Precision: 67.89% 92.45% 70.73% 96.67% 66.92% 77.56% 56.83% 55.42% 48.46%

Table 6: RNN Confusion Matrix. Abbreviations: Lmono(Lm) Lreorder(Lr) Rmono(Rm) Rreorder(Rr)

| Reference | Prediction | Unalign | BEGIN-Rm | BEGIN-Rr | END-Lm | END-Lr | Lm-Rm | Lr-Rm | Lm-Rr | Lr-Rr |
|-----------|------------|---------|----------|----------|--------|--------|-------|-------|-------|-------|
| Unalign   | 589109     | 17299   | 901      | 7870     | 1000   | 639555 | 82413 | 24277 | 3305  |
| BEGIN-Rmono | 1978      | 339686  | 6397     | 0        | 0      | 0      | 0     | 0     | 0     |
| BEGIN-Lmono | 186      | 13812   | 16032    | 0        | 0      | 0      | 0     | 0     | 0     |
| END-Lmono | 2258       | 0       | 0        | 364121   | 4251   | 0      | 0     | 0     | 0     |
| END-Lreorder | 609       | 0       | 0        | 4693     | 8269   | 0      | 0     | 0     | 0     |
| Lmono-Rmono | 142777    | 1       | 0        | 0        | 0      | 4232113| 105266| 78692 | 3294  |
| Lmono-Rreorder | 96278    | 0       | 1        | 0        | 0      | 491989 | 323272| 14635 | 6698  |
| Lreorder-Rreorder | 31118    | 0       | 0        | 0        | 0      | 380483 | 18144 | 198068| 4335  |
| Lreorder-Rreorder | 12366    | 0       | 1        | 0        | 0      | 501212| 25196 | 17098 | 22157 |

Recall: 43.13% 97.59% 53.39% 98.24% 60.53% 92.77% 34.65% 31.33% 17.47%
Precision: 67.19% 94.61% 68.71% 96.60% 61.16% 73.04% 58.32% 59.54% 55.73%

will show later, the model trained with both CRFs and RNN help to improve the translation quality.

Table 5 and Table 6 demonstrate the confusion matrix of the CRFs and RNN errors over the test corpus. The rows represent the correct tag that the classifier should have predicted and the columns are the actually predicted tags. E.g. the number 687724 in first row and first column of Table 5 tells that there are 687724 correctly labeled Unalign tags. The number 15084 in first row and second column of Table 5 represents that there are 15084 Unalign tags labeled incorrectly to Begin-Rmono. Therefore, numbers on the diagonal from the upper left to the lower right corner represent the amount of correctly classified tags and all other numbers show the amount of false labels. The many zeros show that both classifier rarely make mistake for the label "BEGIN-∗" which only occur at the beginning of a sentence. The same is true for the "END-∗" labels.

5.5 Translation Results

Results are summarized in Table 7. Please read the caption for the meaning of abbreviations. An Index column is added for score reference convenience (B for BLEU; T for TER). For the proposed model, significance testing results on both BLEU and TER are reported (B2 and B3 compared to B1, T2 and T3 compared to T1). We perform bootstrap resampling with bounds estimation as described in (Koehn, 2004). The 95% confidence threshold (denoted by ‡ in the table) is used to draw significance conclusions. We add a column avg. to show the average improvements.

From Table 7 we see that the proposed reordering model using CRFs improves the baseline by 0.98 BLEU and 1.21 TER on average, while the proposed reordering model using RNN improves the baseline by 1.32 BLEU and 1.53 TER on average. For line B2 B3 and T2 T3, most scores are better than their corresponding baseline values with more than 95% confidence. The results show that our proposed idea improves the baseline system and RNN trained model performs better than CRFs trained model, in terms of both automatic measure and significance test. To investigate why RNN has lower performance for the tagging task but achieves better BLEU, we build a 3-gram LM on the source side of the training corpus in Table 2 and perplexity values are listed in Table 8. The perplexity of the test corpus for reordering model comparison is much lower than those NIST corpora for translation experiments. In other words, there exists mismatch of the data for reordering model training and actual MT data. This could explain why CRFs is superior to RNN for labeling problem while RNN is better for MT tasks.

For the comparative study, the best method is the tree-based jump model (JUMPTREE). Our proposed model ranks the second position. The difference is tiny: on average only 0.08 BLEU (B3 and B10) and 0.15 TER (T3 and T10). Even with
6 Conclusion

In this paper, a novel tagging style reordering model has been proposed. By our method, the reordering problem is converted into a sequence labeling problem so that the whole source sentence is taken into consideration for reordering decision. By adding an unaligned word tag, the unaligned word phenomenon is automatically implanted in the proposed model. The model is utilized as soft constraints in the decoder. In practice, we do not experience decoding memory increase nor speed slow down.

We choose CRFs and RNN to accomplish the sequence labeling task. The CRFs achieves lower error rate on the tagging task but RNN trained model is better for the translation task. Experimental results show that our model is stable and improves the baseline system by 0.98 BLEU and 1.21 TER (trained by CRFs) and 1.32 BLEU and 1.53 TER (trained by RNN). Most of the scores are better than their corresponding baseline values with more than 95% confidence. We also compare our method with several other popular reordering models. Our model ranks the second position which is slightly worse than the tree-based jump model. However, the tree-based jump model relies on manually designed reordering rules which does not exist for many language pairs while our model can be easily adapted to other translation tasks. We also show that the proposed model is able to improve a very strong baseline system.

The main contributions of the paper are: propose the tagging-style reordering model and improve the translation quality; compare two sequence labeling techniques CRFs and RNN; compare our method with seven other reordering models. To our best knowledge, it is the first time that we propose our method with seven other reordering models. To our best knowledge, it is the first time that we compare our method with seven other reordering models.

Acknowledgments

This work was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation, and also partly based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-12-C-0015. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).
References

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, March.

Adam Berger, Peter F. Brown, Stephen A. Pietra, Vincent J. Pietra, Andrew S. Kehler, and Robert L. Mercer. 1996. Language translation apparatus and method of using Context-Based translation models. *United States Patent*, No. 5,510,981.

Colin Cherry. 2008. Cohesive phrase-based decoding for statistical machine translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL: HLT)*, pages 72–80, Columbus, Ohio, USA, June. Association for Computational Linguistics.

Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.

Minwei Feng, Arne Mauser, and Hermann Ney. 2010. A source-side decoding sequence model for statistical machine translation. In *Proceedings of the Association for Machine Translation in the Americas (AMTA)*, Denver, CO, USA, October.

Minwei Feng, Weiwei Sun, and Hermann Ney. 2012. Semantic cohesion model for phrase-based SMT. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 867–878, Mumbai, India, December. The COLING 2012 Organizing Committee.

Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 848–856, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alex Graves. 2008. *Supervised Sequence Labelling with Recurrent Neural Networks*. Ph.D. thesis, Technical University of Munich, July.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November.

Michael I. Jordan. 1990. Attractor dynamics and parallelism in a connectionist sequential machine. *IEEE Computer Society Neural Networks Technology Series*, pages 112–127.

Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL) - Volume 1*, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, demonstration session, pages 177–180, Prague, Czech Republic, June.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kevin J. Lang, Alex H. Waibel, and Geoffrey E. Hinton. 1990. A time-delay neural network architecture for isolated word recognition. *Neural networks*, 3(1):23–43, January.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513, Association for Computational Linguistics, July.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(3):503–528, December.

José B. Mariño, Rafael E. Banchs, Josep Maria Crego, Adrià de Gispert, Patrik Lambert, José A. R. Fonollosa, and Marta R. Costa-Jussá. 2006. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549, December.

Franz J. Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, Pennsylvania, USA, July.

Franz J. Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP)*, pages 20–28, University of Maryland, College Park, MD, USA, June. Association for Computational Linguistics.
Franz J. Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proceedings of the Conference on Statistical Machine Translation at the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 161–168, Boston, Massachusetts, USA, May. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. *IBM Research Report*, RC22176 (W0109-022), September.

David. E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning internal representations by error propagation. In David E. Rumelhart and James L. McClelland, editors, *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1*, pages 318–362. MIT Press, Cambridge, MA, USA.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, November.

Matthew Snover, Bonnie Dorr, Richard Schwartz, John Makhoul, Linnea Micciulla, and Ralph Weischedel. 2005. A study of translation error rate with targeted human annotation. Technical Report LAMP-TR-126, CS-TR-4755, UMIACS-TR-2005-58, University of Maryland, College Park, MD.

Andreas Stolcke. 2002. Srlm - an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing (ICSLP)*, pages 901–904, Denver, Colorado, USA, September. ISCA.

Charles Sutton and Andrew McCallum, 2006. *Introduction to Conditional Random Fields for Relational Learning*, pages 93–128. MIT Press.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 737–745, Prague, Czech Republic, June. Association for Computational Linguistics.

Dekai Wu. 1995. Stochastic inversion transduction grammars with application to segmentation, bracketing, and alignment of parallel corpora. In *Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI) - Volume 2*, pages 1328–1335, San Francisco, CA, USA, August. Morgan Kaufmann Publishers Inc.

Richard Zens and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL) - Volume 1*, pages 144–151, Stroudsburg, PA, USA. Association for Computational Linguistics.

Richard Zens and Hermann Ney. 2006. Discriminative reordering models for statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation at the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL-HLT)*, pages 55–63, New York City, NY, June. Association for Computational Linguistics.

Richard Zens, Franz J. Och, and Hermann Ney. 2002. Phrase-based statistical machine translation. In *German Conference on Artificial Intelligence*, pages 18–32. Springer Verlag, September.

Yuqi Zhang, Richard Zens, and Hermann Ney. 2007. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *Proceedings of the Workshop on Syntax and Structure in Statistical Translation at the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL-HLT)/Association for Machine Translation in the Americas (AMTA)*, pages 1–8, Morristown, NJ, USA, April. Association for Computational Linguistics.

Yuqi Zhang. 2013. *The Application of Source Language Information in Chinese-English Statistical Machine Translation*. Ph.D. thesis, Computer Science Department, RWTH Aachen University, May.