Sequence Labeling-based Reordering Model for Phrase-based SMT

Minwei Feng, Jan-Thorsten Peter, Hermann Ney

Human Language Technology and Pattern Recognition Group
Computer Science Department
RWTH Aachen University
Aachen, Germany

`<surname>@cs.rwth-aachen.de`

Abstract

For current statistical machine translation system, reordering is still a major problem for language pairs like Chinese-English, where the source and target language have significant word order differences. 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. For the given source sentence, we assign each source token a label which contains the reordering information for that token. We also design an unaligned word tag so that the unaligned word phenomenon is automatically implanted in the proposed model. Our reordering model is conditioned on the whole source sentence. Hence it is able to catch the long dependency in the source sentence. Although the learning on large scale task requests notably amounts of computational resources, the decoder makes use of the tagging information as soft constraints. Therefore, the training procedure of our model is computationally expensive for large task while in the test phase (during translation) our model is very efficient. We carried out experiments on five Chinese-English NIST tasks trained with BOLT data. Results show that our model improves the baseline system by 1.32 BLEU 1.53 TER on average.

1. Introduction

The systematic word order difference between two languages, pose 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 [1], if arbitrary reordering is allowed, the search problem is NP-hard.

In this paper, we propose a novel tagging style reordering model. Our model converts the reordering problem into a sequence labeling problem, i.e. a tagging task. For the given source sentence, we assign each source token a label which contains the reordering information for that token. We also design an unaligned word tag so that the unaligned word phenomenon is automatically implanted in the proposed model. Our model is conditioned on the whole source sentence. Hence it is able to capture the long dependency in the source sentence. We compare two training methods: conditional random fields (CRFs) and recurrent neural network (RNN). Although the learning on large scale task requests notably amounts of computational resources, the decoder makes use of the tagging information as soft constraints. Therefore, the training procedure of our model is computationally expensive while in the test phase (during translation) our model is very efficient.

The remainder of this paper is organized as follows: Section 2 reviews the related work for solving the reordering problem. Section 3 introduces the basement of this research: the principle of statistical machine translation. Section 4 describes the proposed model. Section 5 provides the experimental configuration and results. Conclusion will be given in Section 6.

2. Related Work

Many ideas have been proposed to address the reordering problem. Early work focuses on reordering constraints, e.g. using ITG constraints [2] and IBM constraints [3] to model the sequence permutation. Within the phrase-based SMT framework there are mainly three stages where improved reordering could be integrated:

1. Reorder the source sentence. So that the word order of source and target sentences is similar. Usually it is done as the preprocessing step for both training data and test data.
2. In the decoder, add models in the log-linear framework or constraints in the decoder to reward good reordering options or penalize bad ones.
3. In the reranking framework.

For the first point, [4] used manually designed rules to reorder parse trees of the source sentences as a preprocessing step. Based on shallow syntax, [5] used 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 second category, researchers try to inform the decoder on what a good reordering is or what a suitable decoding sequence is. [6] used a discriminative reordering model to predict the orientation of the next phrase given the previous phrase. [7] presents 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 to find the suitable order of translation units. [8] puts the syntactic cohesion as a soft constraint in the decoder to guide the decoding process to choose those translations that do not violate the syntactic structure of the source sentence. Adding new features in the log-linear framework has the advantage that the new feature has access to the whole search space. Another advantage of methods in this category is that we let the decoder decide the weights of features, so that even if one model gives wrong estimation sometimes, it can still be corrected by other models. Our work in this paper belongs to this category.

In the reranking step, the system has the last opportunity to choose a good translation. [9] describe the use of syntactic features in the rescoring step. They report the most useful feature is IBM Model 1 score. The syntactic features contribute very small gains. Another disadvantage of carrying out reordering in reranking is the representativeness of the N-best list is often a question mark.

3. Translation System Overview

In this section, we are going to describe the phrase-based SMT system we used for the experiments.

In statistical machine translation, we are given a source language sentence $f_1^J = f_1 \dots f_j \dots f_J$. The objective is to translate the source into a target language sentence $e_1^I = e_1 \dots e_i \dots e_I$. The strategy is among all possible target language sentences, we will choose the one with the highest probability:

$$\hat{e}_i^{\hat{I}} = \arg\max_{I,e_1^I} \{Pr(e_1^I|f_1^J)\} \qquad (1)$$

We model $Pr(e_1^I|f_1^J)$ directly using a log-linear combination of several models [10]:

$$Pr(e_1^I|f_1^J) = \frac{\exp\Big(\sum\limits_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J)\Big)}{\sum\limits_{I',e'^{I'}_1} \exp\Big(\sum\limits_{m=1}^{M} \lambda_m h_m(e'^{I'}_1, f_1^J)\Big)} \qquad (2)$$

The denominator is to make the $Pr(e_1^I|f_1^J)$ to be a probability distribution and it depends only on the source sentence f_1^J. For search, the decision rule is simply:

$$\hat{e}_i^{\hat{I}} = \arg\max_{I,e_1^I} \Big\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J) \Big\} \qquad (3)$$

The model scaling factors λ_1^M are trained with Minimum Error Rate Training (MERT).

In this paper, the phrase-based machine translation system is utilized [11, 12, 13]. The translation process consists in segmenting the source sentence according to the phrase table which is built from the word alignment. The translation of each of these segments consists in just extracting the target side from the phrase pair. With the corresponding target side, the final translation is the composition of these translated segments. In this last step, reordering is allowed.

4. Tagging-style Reordering Model

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

4.1. 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_1^7, the system translates it into the target sentence e_1^7. The alignment link set $\{a_1 = 3, a_3 = 2, a_4 = 4, a_4 = 5, a_5 = 7, a_6 = 6, a_7 = 6\}$ 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 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.

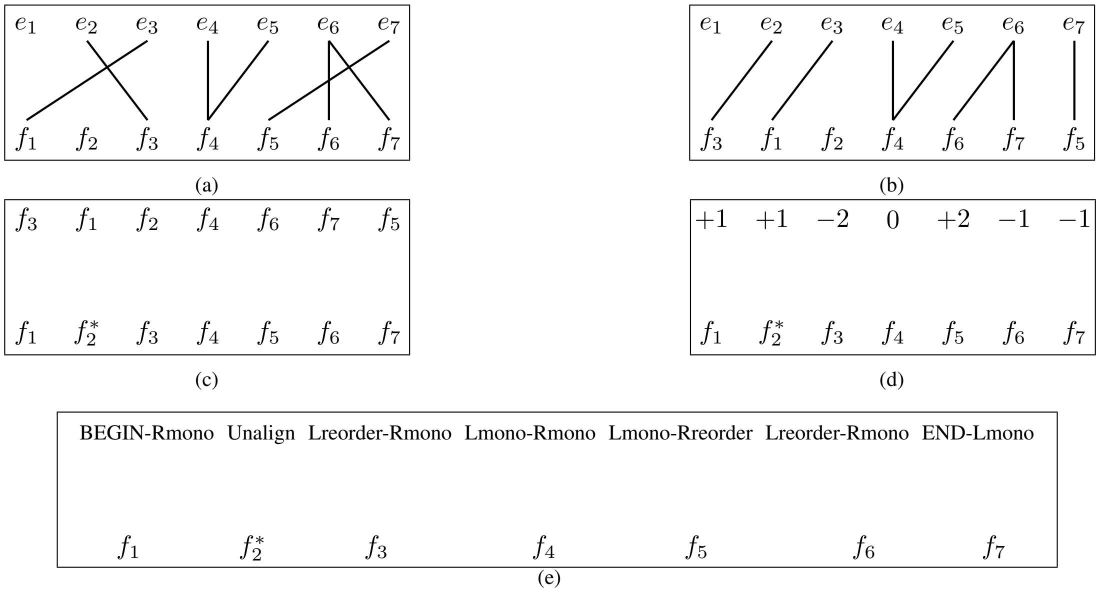

Figure 1: Modeling process illustration.

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 has the exactly same length. Hence the relation can be modeled by a function $F(f)$ which assigns a value for each of the source word f. Figure 1(d) manifests this function. The positive function values mean that compared to the original position in the source sentence, its position in the decoding sequence should move right. If the function value is 0, the word's position in original source sentence and its decoding sequence is same. For example, f_1 is the first word in the source sentence but it is the second word in the decoding sequence. So its function value is $+1$ (move right one position).

Now Figure 1(d) converts the reordering problem into a sequence labeling or tagging problem. To make the computational cost to a reasonable level, we do a final step simplification in Figure 1(e). Suppose the longest sentence length is 100, then according to Figure 1(d), there are 200 tags (from -99 to +99 plus the unalign tag). As we will see later, this number is too large for our task. We instead design nine tags. For a source word f_j in one source sentence f_1^J, the tag of f_j will be one of the following:

Unalign f_j is an unaligned source word

BEGIN-Rmono $j = 1$ and f_{j+1} is translated *after* f_j (Rmono for right monotonic)

BEGIN-Rreorder $j = 1$ and f_{j+1} is translated *before* f_j (Rreorder for right reordered)

END-Lmono $j = J$ and f_{j-1} translated *before* f_j (Lmono for left monotonic)

END-Lreorder $j = J$ and f_{j-1} translated *after* f_j (Lreorder for left reordered)

Lmono-Rmono $1 < j < J$ and f_{j-1} translated *before* f_j and f_j translated *before* f_{j+1}

Lreorder-Rmono $1 < j < J$ and f_{j-1} translated *after* f_j and f_j translated *before* f_{j+1}

Lmono-Rreorder $1 < j < J$ and f_{j-1} translated *before* f_j and f_j translated *after* f_{j+1}

Lreorder-Rreorder $1 < j < J$ and f_{j-1} translated *after* f_j and f_j translated *after* f_{j+1}

Up to this point, we have converted the reordering problem into a tagging problem with nine tags. The transformation in Figure 1 is conducted for all the sentence pairs in the bilingual training corpus. After that, we have built an "annotated" corpus for the training. For this supervised learning task, we choose the approach conditional random fields (CRFs) [14, 15, 16] and recurrent neural network (RNN) [17, 18, 19].

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) [20, 21], 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) [22] which reads the input sequence from both directions before making the prediction. The long short-term memory (LSTM) [23] 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 [24].

4.2. Decoding

Once the model training is finished, we make inference on develop and test corpora. After 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_1^J, e_1^I) will be formally splitted into a segmentation S_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 kth target phrase $\tilde{e}_k$. The start and end position of the kth source phrase $\tilde{f}_k$ are b_k and j_k. Suppose the search state is now extended with a new phrase pair $(\tilde{f}_k, \tilde{e}_k)$:

$$\tilde{f}_k := f_{b_k} \dots f_{j_k} \tag{4}$$

$$\tilde{e}_k := e_{i_{k-1}+1} \dots e_{i_k} \tag{5}$$

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 consistence 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.

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 (Table 1) from the BOLT task
  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 [25] with modified Kneser-Ney smoothing
  training data: target side of bilingual data.
- BLEU [26] and TER [27] reported
  all scores calculated in lowercase way.
- Wapiti toolkit [16] used for CRFs; RNN is built by the RNNLIB [28] toolkit.

	Chinese	**English**
Sentences	5 384 856	
Running Words	115 172 748	129 820 318
Vocabulary	1 125 437	739 251

Table 1: 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 2. From the figure we can see that most words are monotonic. We then divide the corpus to three parts: train, validation and test. The source side data statistics for the reordering model training is given in Table 2 (target side has only nine labels).

	train	**validation**	**test**
Sentences	2 973 519	400 000	400 000
Running Words	62 263 295	8 370 361	8 382 086
Vocabulary	454 951	149686	150 007

Table 2: reordering model training data statistics

5.2. CRFs Training Results

The toolkit Wapiti [16] is used in this paper. We choose the classical optimization algorithm limited memory BFGS (L-BFGS) [29]. For regularization, Wapiti uses both the ℓ^1 and ℓ^2 penalty terms, yielding the elastic-net penalty of the form

$$\rho_1 \cdot \parallel \theta \parallel_1 + \frac{\rho_2}{2} \cdot \parallel \theta \parallel_2^2 \qquad (6)$$

In this work, we use as many features as possible because ℓ^1 penalty $\rho_1 \parallel \theta \parallel_1$ is able to yield sparse parameter vectors, i.e. using a ℓ^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 ρ_1 and ρ_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. The feature template we set initially will generate 722 999 637 features. After training 36 902 363 features are kept.

5.3. RNN Training Results

We also applied RNN to the task as an alternative approach to CRFs. The here used RNN implementation is RNNLIB [28] which has support for long short term memory (LSTM) [30]. 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 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. RNNLIB 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

From machine learning point of view, 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 will show later, the model trained with both CRFs and RNN help to improve the translation quality.

Table 3 and Table 4 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 nunber 687724 in first row and first column of Table 3 tells that there are 687724 correctly labeled **Unalign** tags. The number 15084 in first row and second column of Table 3 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 6. Automatic measure BLEU and TER scores are provided. Also we report significance testing results on both BLEU and TER. We perform bootstrap resampling with bounds estimation as described in [31]. We use the 95% confidence threshold (denoted by ‡ in the table) to draw significance conclusions. Besides the five test corpora, we add a column **avg.** to show the average improvements. We also add a column **Index** for score reference convenience.

From Table 6 we see that our 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 the CRFs-based model, the largest BLEU improvement 1.15 is from NIST05 and the largest TER improvement 1.57 is from NIST03. The improvements are even larger with the tags created by the RNN with a BLEU improvement of 1.70 and a TER improvement 1.98 for NIST02. For line 3 and 6, all the scores are better than their corresponding baseline values with more than 95% confidence. For line 2 and 5, three out of the five 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 5-gram LM on the source side of the training corpus in Table 2. Perplexity values are provided in Table 5. We see clearly that 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.

	Running Words	**OOV**	**Perplexity**
Test in Table 2	8 382 086	0	6.665
NIST02	22 749	391	234.494
NIST03	24 180	518	346.242
NIST04	49 612	700	223.492
NIST05	29 966	511	342.925
NIST08	32 502	998	473.975

Table 5: Perplexity

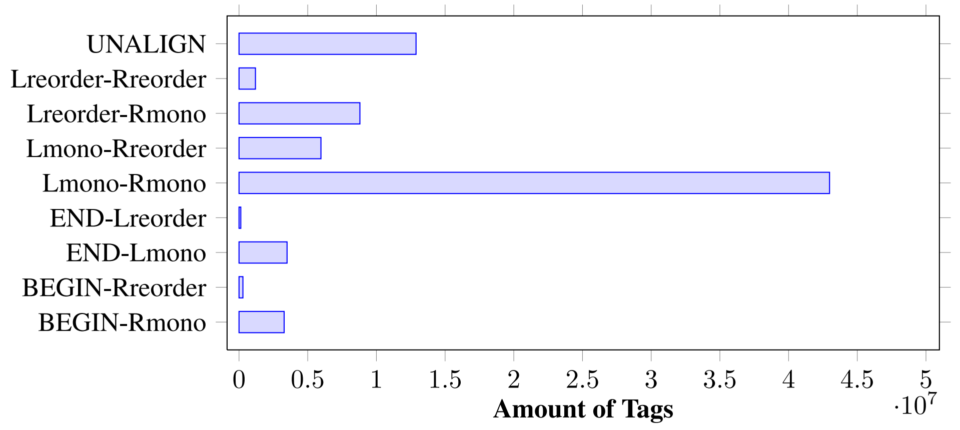

Figure 2: Tags distribution illustration.

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	3537	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	7913	0	0	0	0
Lmomo-Rmono	188627	0	0	0	0	4032738	176682	150952	13114
Lreorder-Rmono	88177	0	0	0	0	369232	433027	27162	15275
Lmomo-Rreorder	32342	0	0	0	0	268570	24558	296033	10645
Lreorder-Rreorder	9865	0	0	0	0	34746	20382	16514	45342
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 3: 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	589100	17299	901	7870	1000	639555	82413	24277	3305
BEGIN-Rmono	1978	339686	6397	0	0	0	0	0	0
BEGIN-Rreorder	186	13812	16032	0	0	0	0	0	0
END-Lmono	2258	0	0	364121	4251	0	0	0	0
END-Lreorde	699	0	0	4693	8269	1	0	0	0
Lmomo-Rmono	142777	1	0	0	0	4232113	105266	78692	3264
Lreorder-Rmono	96278	0	1	0	0	491989	323272	14635	6698
Lmomo-Rreorder	31118	0	0	0	0	380483	18144	198068	4335
Lreorder-Rreorder	12366	0	1	0	0	50121	25196	17008	22157
Recall	43.13%	97.59%	53.39%	98.24%	60.53%	92.77%	34.65%	31.33%	17.47%
Precision	67.19%	91.61%	68.71%	96.66%	61.16%	73.04%	58.32%	59.54%	55.73%

Table 4: RNN Confusion Matrix. Abbreviations: Lmono(Lm) Lreorder(Lr) Rmono(Rm) Rreorder(Rr)

6. Conclusion

In this paper, a novel tagging style reordering model has been proposed. By our modeling 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. Although the training phase of our model needs large computational costs, its usage for decoding is quite simple. 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 learning task takes huge amount of features and significant computational costs. Both ℓ^1 and ℓ^2 penalty are used in regularization. Hence the feature selection is automatically conducted. For test corpus, the token error rate is 25.75% and the sequence error rate is 69.39%. For

Systems	NIST02	NIST03	NIST04	NIST05	NIST08	avg.	Index
			BLEU scores				
baseline	33.60	34.29	35.73	32.15	26.34	-	1
baseline+CRFs	34.53	35.19	36.56‡	33.30‡	27.41‡	0.98	2
baseline+RNN	35.30‡	35.34‡	37.03‡	33.80‡	27.23‡	1.32	3
			TER scores				
baseline	61.36	60.48	59.12	60.94	65.17	-	4
baseline+CRFs	60.14‡	58.91‡	57.91‡	59.77‡	64.30‡	1.21	5
baseline+RNN	59.38‡	58.87‡	57.60‡	59.56‡	63.99‡	1.53	6

Table 6: Experimental results. ‡ means the value is better than its corresponding baseline with more than 95% confidence.

RNN training, we adopt the bidirectional RNN with LSTM. For test corpus, the token error rate is 27.31% and the sequence error rate is 77.00%.

We utilize our model as soft constraints in the decoder. 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.

The two main contributions are: propose the tagging-style reordering model and prove its ability to improve the translation quality; compare two sequence labeling techniques CRFs and RNN. To our best knowledge, this is the first experimental comparison of the CRFs and RNN.

7. Acknowledgements

This work was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. 4911028154.0. 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).

8. References

[1] K. Knight, "Decoding complexity in word-replacement translation models," *Comput. Linguist.*, vol. 25, no. 4, pp. 607–615, 1999.

[2] D. Wu, "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 - Volume 2*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 1328–1335.

[3] A. Berger, P. F. Brown, S. A. Pietra, V. J. Pietra, A. S. Kehler, and R. L. Mercer, "Language translation apparatus and method of using context-based translation models," United States Patent 5510981, April 1996.

[4] C. Wang, M. Collins, and P. Koehn, "Chinese syntactic reordering for statistical machine translation," in *Proceedings of the EMNLP/CoNLL-07*, Prague, Czech Republic, June 2007, pp. 737–745.

[5] Y. Zhang, R. Zens, and H. Ney, "Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation," in *Proceedings of the NAACL-HLT-07/AMTA Workshop on Syntax and Structure in Statistical Translation*, Morristown, NJ, USA, April 2007, pp. 1–8.

[6] R. Zens and H. Ney, "Discriminative reordering models for statistical machine translation," in *Proceedings of the Workshop on Statistical Machine Translation at HLT-NAACL-06*, New York City, NY, June 2006, pp. 55–63.

[7] J. B. Mariño, R. E. Banchs, J. M. Crego, A. de Gispert, P. Lambert, J. A. R. Fonollosa, and M. R. Costa-Jussà, "*N*-gram-based machine translation," *Computational Linguistics*, vol. 32, no. 4, pp. 527–549, 2006.

[8] C. Cherry, "Cohesive phrase-based decoding for statistical machine translation," in *Proceedings of ACL-08: HLT*, Columbus, Ohio, June 2008, pp. 72–80.

[9] F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev, “A smorgasbord of features for statistical machine translation,” in *Proceedings of NAACL-HLT-04*, Boston, Massachusetts, USA, May 2004, pp. 161–168.

[10] F. J. Och and H. Ney, “Discriminative training and maximum entropy models for statistical machine translation,” in *Proceedings of ACL-02*, Philadelphia, Pennsylvania, USA, July 2002, pp. 295–302.

[11] F. J. Och, C. Tillmann, and H. Ney, “Improved alignment models for statistical machine translation,” in *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP99)*, University of Maryland, College Park, MD, USA, June 1999, pp. 20–28.

[12] R. Zens, F. J. Och, and H. Ney, “Phrase-based statistical machine translation,” in *German Conference on Artificial Intelligence*. Springer Verlag, 2002, pp. 18–32.

[13] P. Koehn, F. J. Och, and D. Marcu, “Statistical phrase-based translation,” in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, ser. NAACL ’03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 48–54.

[14] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML ’01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 282–289.

[15] C. Sutton and A. Mccallum, *Introduction to Conditional Random Fields for Relational Learning*. MIT Press, 2006.

[16] T. Lavergne, O. Cappé, and F. Yvon, “Practical very large scale CRFs,” in *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, July 2010, pp. 504–513.

[17] J. L. Elman, “Finding structure in time,” *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[18] M. I. Jordan, “Attractor dynamics and parallelism in a connectionist sequential machine,” *IEEE Computer Society Neural Networks Technology Series*, pp. 112–127, 1990.

[19] K. J. Lang, A. H. Waibel, and G. E. Hinton, “A time-delay neural network architecture for isolated word recognition.” *Neural networks*, vol. 3, no. 1, pp. 23–43, 1990.

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Parallel distributed processing: explorations in the microstructure of cognition, vol. 1,” D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA, USA: MIT Press, 1986, ch. Learning internal representations by error propagation, pp. 318–362.

[21] C. M. Bishop, *Neural Networks for Pattern Recognition*, 1st ed. Oxford University Press, USA, Jan. 1996.

[22] M. Schuster and K. Paliwal, “Bidirectional recurrent neural networks,” *Trans. Sig. Proc.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1109/78.650093

[23] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[24] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult.” *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[25] A. Stolcke, “Srilm - an extensible language modeling toolkit,” in *Proceedings of ICSLP-02*, Denver, Colorado,USA, September 2002, pp. 901–904.

[26] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” no. RC22176 (W0109-022), September 2001.

[27] M. Snover, B. Dorr, R. Schwartz, J. Makhoul, L. Micciulla, and R. Weischedel, “A study of translation error rate with targeted human annotation,” University of Maryland, College Park, MD, Tech. Rep. LAMP-TR-126, CS-TR-4755, UMIACS-TR-2005-58, 2005.

[28] A. Graves, “Rnnlib: A recurrent neural network library for sequence learning problems,” http://sourceforge.net/projects/rnnl/.

[29] D. C. Liu and J. Nocedal, “On the limited memory bfgs method for large scale optimization,” *Math. Program.*, vol. 45, no. 3, pp. 503–528, Dec. 1989.

[30] A. Graves, “Supervised Sequence Labelling with Recurrent Neural Networks,” Ph.D. dissertation, 2008.

[31] P. Koehn, “Statistical significance tests for machine translation evaluation,” Barcelona, Spain, July 2004, pp. 388–395.