Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Translation Prediction with Source Dependency-Based Context Representation

Kehai Chen,[1] Tiejun Zhao,[1*] Muyun Yang,[1] Lemao Liu[2]
[1]Machine Intelligence and Translation Laboratory, Harbin Institute of Technology, Harbin, China
[2]ASTREC, National Institute of Information and Communications Technology, Kyoto, Japan
{khchen, tjzhao, yangmuyun}@hit.edu.cn, lmliu@nict.go.jp

Abstract

Learning context representations is very promising to improve translation results, particularly through neural networks. Previous efforts process the context words sequentially and neglect their internal syntactic structure. In this paper, we propose a novel neural network based on bi-convolutional architecture to represent the source dependency-based context for translation prediction. The proposed model is able to not only encode the long-distance dependencies but also capture the functional similarities for better translation prediction (i.e., ambiguous words translation and word forms translation). Examined by a large-scale Chinese-English translation task, the proposed approach achieves a significant improvement (of up to +1.9 BLEU points) over the baseline system, and meanwhile outperforms a number of context-enhanced comparison system.

1 Introduction

Learning context representation for translation prediction has attracted much attention in statistical machine translation (SMT). Many research works model the target context as monolingual language models for SMT (Shen, Xu, and Weischedel 2008; Mikolov 2012; Vaswani et al. 2013). Recently, the main focus is shifting from a monolingual context to a bilingual context, for example, the bilingual language model (BiLM) (Crego and Yvon 2010; Niehues et al. 2011; Garmash and Monz 2014) and operation sequence model (OSM) based on Minimum Translation Units (MTU) (Durrani, Schmid, and Fraser 2011; Durrani et al. 2013). However, their works relied on the traditional n-gram method, which is limited to relatively small windows and lacks generalization abilities in semantics (Mikolov et al. 2013) due to data sparsity (Guta et al. 2015). On the other hand, many efforts have been made on learning representations of bilingual context with neural networks (Auli et al. 2013; Liu et al. 2013; Hu et al. 2014; Devlin et al. 2014).

Auli et al. and Hu et al. employed recurrent neural networks to represent the bilingual context either in word level or in phrase (i.e., MTU) level for SMT. However, their models are used for post-processing via n-best rescoring instead of translation decoding, because of the recurrence on the target side. Most notably neural network joint model (NNJM) described in (Devlin et al. 2014), which encodes n-gram words in target side and "relevant" words in the source side using a feed-forward neural network (FFNN), breaks down the recurrence on the target side and thus can be integrated into translation decoding. However, these models can not capture long-distance dependencies among the source-side words, due to the nature of window-based FFNN[1].

More recently, Meng et al. and Zhang, Zhang, and Hao modeled long-distance dependencies by encoding the entire source sentence with neural networks. Meng et al. achieved this by using Convolutional Neural Networks (CNN) to summarize informative features from source sentences, but their method only used the sentences shorter than 40 words for faster training, due to the variant length of sentences. Zhang, Zhang, and Hao proposed a chunk-based counterpart for this issue, in which the number of chunks without linguistic guarantee needs be predefined and tuned as an additional hyperparameter. Furthermore, these methods represent the entire source sentence as a fixed vector for translations at different time steps rather than a dynamic vector as the attention mechanism in (Bahdanau, Cho, and Bengio 2015; Luong, Pham, and Manning 2015; Liu et al. 2016).

In this paper, we propose an alternative approach to capturing long-distance dependencies with clues from a source-side dependency tree, inspired by the success of work on encode sentences non-sequentially (Levy and Goldberg 2014; Tai, Socher, and Manning 2015; Eriguchi, Hashimoto, and Tsuruoka 2016). The proposed model is able to be integrated into decoding; also, it makes use of the source-side long-distance dependencies, and learns the dynamical context representation for translation prediction at different time steps. In particular, instead of encoding the source words sequentially, it encodes the context structurally depending on source-side dependency tree. This can not only encode the long-distance dependencies for better ambiguous words translation (Chan, Ng, and Chiang 2007; Carpuat and Wu 2007), but also capture the functional similarities for better word forms translation (Levy and Goldberg 2014), which

*Corresponding author.

[1]It is possible to enlarge the window to cover the entire source sentence, but this does not lead to further improvements as shown in (Devlin et al. 2014).

is verified in our experiments (§5.5). To this end, firstly an aligned bilingual parallel sentence pair with source-side dependency is converted into a novel dependency-based bilingual compositional sequence (DBiCS) (§2); then a unique bi-convolutional neural network (BiCNN) is presented to learn semantic representations for DBiCS units (§3); and finally a DBiCS based neural network language model (DBiCSNNLM) is further designed to predict translation based on the learned representations (§4).

This paper makes the following contributions:

- It proposes a novel neural network based approach to encoding context with source-side dependency information. Instead of sequentially encoding the source words, we encode them in a dependency structure. The proposed model can encode the long-distance dependencies and capture the functional information for better translation prediction (i.e., ambiguous words and word forms translation).
- Intensive experiments on NIST Chinese-English translation tasks show that the proposed DBiCSNNLM achieves significant and substantial improvements (of 1.9 BLEU points on average) over the baseline system, and meanwhile advances a number of methods on bilingual context representation, particularly including NNJM (§5).

2 Dependency-Based Bilingual Compositional Sequence (DBiCS)

Suppose an aligned bilingual sentence is represented as $\langle \mathbf{f} = f_1^J, \mathbf{e} = e_1^I, A \rangle$, where J (I) denotes the source (target) length and A denotes the alignment between the sentence $\mathbf{f}$ and $\mathbf{e}$. In order to make the proposed approach simple, the alignment A is preprocessed to suffice that, each target word is aligned to one source word as shown in Figure (1), i.e., $A = \{a_1, \cdots, a_I\}$ with $a_i \in \{1, \cdots, J\}$. In addition, the dependency tree of a sentence $\mathbf{f}$ is denoted as $T_{\mathbf{f}}$.

A dependency-based bilingual compositional sequence (**DBiCS**) is defined upon an aligned bilingual sentence $\langle \mathbf{f}, \mathbf{e}, A \rangle$ and its source dependency tree $T_{\mathbf{f}}$. A DBiCS is an ordered sequence of tuples, called dependency-based bilingual compositional unit (**DBiCU**), which is used to segment the bilingual sentence with the order respecting to $\mathbf{e}$ while retaining the dependencies in $T_{\mathbf{f}}$. In this paper, we employ the minimum translation units to segment the aligned bilingual sentence following (Durrani, Schmid, and Fraser 2011). In addition, although it is possible to consider all kinds of dependencies from the dependency tree, we only keep three of them for simplicity, which are parent, siblings, and children. In this way, each DBiCU is corresponding to a MTU augmented with three kinds of dependencies.

Formally, suppose $\langle M_{\mathbf{e}}^i, M_{\mathbf{f}}^i \rangle$ denotes the i_{th} MTU for $\langle \mathbf{f}, \mathbf{e}, A \rangle$, then its corresponding DBiCU U_i is defined as the following tuple of words:

$$U_i = \langle M_{\mathbf{e}}^i \parallel PA, SI, M_{\mathbf{f}}^i, CH \rangle, \quad (1)$$

where PA, SI, CH denote the parent, siblings and children words of $M_{\mathbf{f}}^i$ in a dependency tree; '$\parallel$' is used to split the words in a DBiCU into the target part and the source part.

For example, in Figure 1, the first MTU is $\langle$today, *jintian*$\rangle$. Then in the source-side dependency tree T_f, the parent, siblings and children of "*jintian*" are "*gongbu*", "*zhongguo*, *le*, *chengguo*", and "ε" (no child), which are respectively marked in blue, green and purple in Figure 2. Therefore, the corresponding DBiCU $\langle$today$\|$*gongbu*, *zhongguo*, *le*, *chengguo*, *jintian*, $\varepsilon\rangle$, as shown in Figure 2.

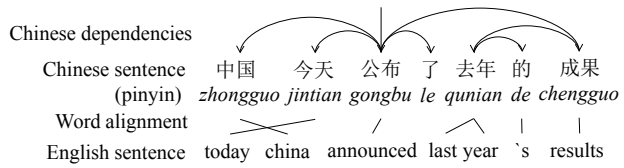

Figure 1: Chinese-English parallel sentence pair with Chinese dependencies and word alignment.

Our DBiCU is defined on a minimum translation unit and thus it is closely related to (Hu et al. 2014), but ours encodes the dependency structure from the source side dependency tree instead of sequential words. Moreover, our DBiCS is similar to that in (Devlin et al. 2014), where an unit is a word pair $\langle e_i, f_{a_i} \rangle$ augmented with n-gram contexts of both e_i and f_{a_i}. Note that in order to highlight the contribution of the source dependency context in this paper, our DBiCS excludes both n-gram contexts of M_e and M_f, even if they might lead to further improvements.

In summary, given a tuple $\langle \mathbf{f}, \mathbf{e}, A, T_{\mathbf{f}} \rangle$, we can obtain its unique DBiCS as follows: we firstly identify the ordered MTU sequence according to A; and then we can convert each MTU to its corresponding DBiCU by looking up the dependencies in $T_{\mathbf{f}}$ and thus get the interested DBiCS. For the example in Figure 1, we can obtain its DBiCS as shown in Figure 2, which consists of 6 ordered DBiCUs.

3 BiCNN Architecture for DBiCU

This section will introduce a new variant of the standard convolutional neural network (CNN) (Collobert et al. 2011) called the Bi-Convolutional neural network (**BiCNN**). The BiCNN consists of two interactional CNNs, and each of them processes either a source-side or a target-side part of DBiCU. Learning semantic representation for each DBiCU considers mutual influence between source-side and target-side part of DBiCU, thus alleviating unconformity between the bilingual vector space, as shown in Figure 3.

Input Layer: The input layer includes two matrices for a DBiCU U. One matrix is $U_s = \{w_1, ..., w_i\}$ for the source-side part of U, the other matrix is $U_t = \{w_1, ..., w_j\}$ for the target-side part of U. Each of them is a matrix $C^{n \times d}$, d is the dimension of word embedding, n is the length of U_s or U_t[2]. For example, a DBiCU $\langle$last year$\|$*chengguo*, ε, *qunian*, *de*$\rangle$, whose source-side part and target-side part are less than 10 words, is converted into a new DBiCU $\langle$/, /, /, /, /, /, /, /, /, last, year $\|$/, /, /, /, /, /, /,*chengguo*, ε, *qunian*, *de*$\rangle$ after padding "/", which denotes a special word. In the paper, two hyperparameters are set $n = 10, d = 100$,

[2]We find that source-side and target-side length of 99% DBiCU are less than 10 words. So we filter the DBiCUs whose source-side or target-side is greater than 10.

Figure 2: Dependency-based bilingual compositional sequence of Figure 1.

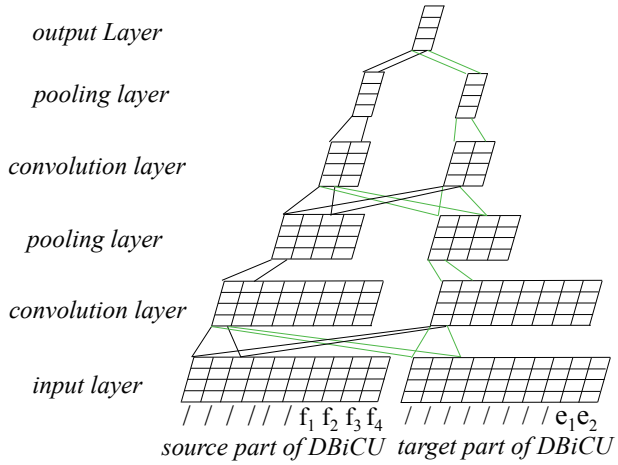

Figure 3: The Bi-Convolutional neural network (BiCNN) for learning semantic representation of DBiCU.

source-side and target-side of a DBiCU are represented as a matrix of the dimension 10×100, respectively.

Convolutional Layer A convolutional layer in the network contains two filters $W_m \in R^{d \times k}$, and $m = \{0, 1\}$, corresponding to the source-side context and the target-side context respectively. Let the filter window size be t (e.g., t=3), the filter W_m generates the feature y_i^m as follows:

$$y_k^m = \sigma(W_m([w_i + w_{i+1} + w_{i+2}] + [w_j + w_{j+1} + w_{j+2}]) + b) \quad (2)$$

where σ is a non-linear activation function (e.g. Relu), and b is a bias term. After the filter traverses the input matrix from w_i to w_{i+t-1} and also from w_j to w_{j+t-1} (1≤i≤n-t+1 and 1≤j≤n-t+1), the output of the feature map y^m is:

$$y^m = [y_1^m, y_2^m, ..., y_{n-t+1}^m] \quad (3)$$

We will denote the feature maps of source-side context and target-side context by y^0 and y^1, respectively.

Pooling Layer: The pooling operation (*max, average*, etc.) is commonly used to extract robust features from convolution. For the output feature map of the convolution layers, column-wise *max* over windows of t=2 consecutive columns is performed (Zhang, Zhang, and Hao 2015):

$$p_i^m = max[y_{2i-1}^m, y_{2i}^m] \quad (4)$$

After the max pooling traverses each window from y_{2i}^m to y_{2i+1}^m, 1≤i≤n/2-t+1, the output of the feature map p^m is:

$$p^m = [p_1^m, p_2^m, ..., y_{n/2-t+1}^m] \quad (5)$$

Then the pooling layer output are p^0 and p^1, respectively.

Output Layer: The output layer is typically a fully connected layer multiplied by a matrix. In the paper, row-wise averaging from pooling layers is performed without any parameters for simplicity, whereby the semantic representation V of a DBiCU is a vector obtained:

$$V = average(p_0 + p_1) \quad (6)$$

where p_0 and p_1 are the output of the second pooling layer.

Therefore, the above BiCNN plays a role of function φ parameterized by θ_1, which maps a DBiCU U into V:

$$V = \varphi(U; \theta_1) \quad (7)$$

Note that V in Eq.(6) is a vector rather than a scalar, which will be transformed into a scalar to score a translation later.

4 Translation Prediction with DBiCS Representation

In the section, we will firstly propose the entire model based on DBiCS to score a translation hypothesis, and then present the training process of the proposed model.

4.1 DBiCSNNLM

The proposed model can be illustrated in Figure 4 (left), which mainly consists of two components: 1) an inlayer BiCNN as presented in the last section is used to learn semantic representations for the DBiCUs in a DBiCS, and 2) an outlayer FFNN is used to predict the next DBiCU given the previous DBiCUs.

Given a $\mathbf{f}$ and its dependency tree $T_{\mathbf{f}}$, for any translation $\mathbf{e}$ with alignment A, we can obtain its corresponding DBiCS denoted as $\{U_1, \cdots, U_l\}$ with length l. Then we define the following model to score $\langle \mathbf{f}, \mathbf{e}, A, T_{\mathbf{f}} \rangle$:

$$\prod_{i=4}^{l} P(U_i \mid U_{i-1}, U_{i-2}, U_{i-3}; \theta) = \prod_{i=4}^{l} \frac{\exp\left(\phi(V_i, V_{i-1}, V_{i-2}, V_{i-3}; \theta_2)\right)}{Z(U_{i-1}, U_{i-2}, U_{i-3}; \theta)} \quad (8)$$

with $Z(U_{i-1}, U_{i-2}, U_{i-3}; \theta)$ as the normalization:

$$\sum_{\bar{U}} \exp\left(\phi(\bar{V}, V_{i-1}, V_{i-2}, V_{i-3}; \theta_2)\right) \quad (9)$$

where $V{=}\varphi(U; \theta_1)$ is defined as in Eq.(7); ϕ is a feedforward neural network parameterized by θ_2; and $\theta{=}\langle \theta_1, \theta_2 \rangle$ denotes all the model parameters including those in both BiCNN and feedforward neural networks. Since the BiCNN

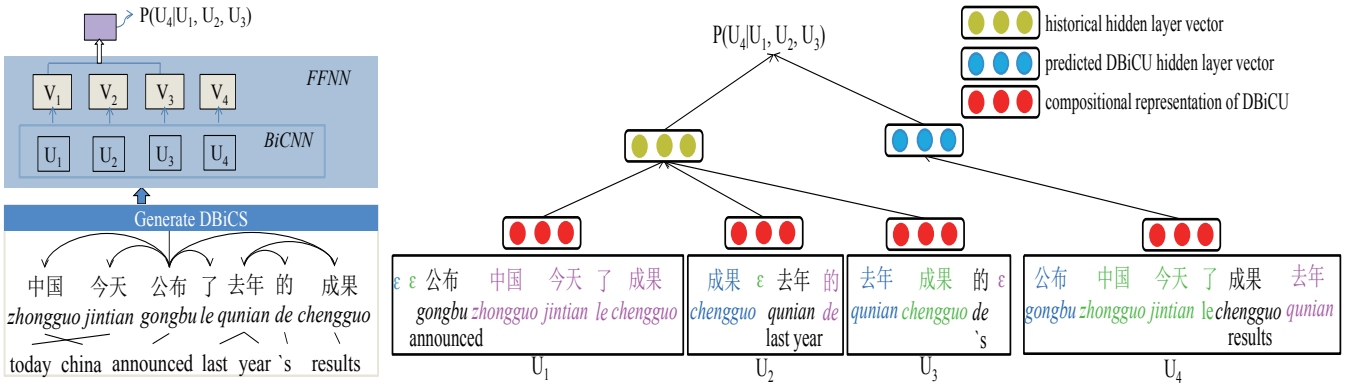

Figure 4: The architecture of DBiCSNNLM (left) and an example of DBiCSNNLM (right).

part learns semantic representations for a DBiCS and the feedforward part is factorized over 4 DBiCUs similar to an n-gram language model, the model is called the DBiCS based neural network language model, **DBiCSNNLM**.

Figure 4 (right) is used as an example of how the presented DBiCSNNLM works on the DBiCS. If the target translation hypothesis "*announced last year 's result*" is evaluated, its DBiCS is shown $\{U_1, U_2, U_3, U_4\}$ in Figure 4 (right). When the predicted DBiCU is U_4 and the preceding historical DBiCUs (assume 4-gram) are $\{U_1, U_2, U_3\}$, the prediction process is finished by two steps: First, the in-layer BiCNN inside the DBiCSNNLM is used to learn semantic representation for each DBiCU U_i (i.e., U_1, U_2, U_3, and U_4). Then these semantic representations are treated as specific context representations at the current time step, by which the outlayer FFNN can give the conditional probability for the predicted DBiCU U_4, e.g., $P(U_4|U_1, U_2, U_3)$.

Since Eq.(8) is factorized over the DBiCUs, it is straightforward to incrementally calculate the DBiCSNNLM scores of a partial translation hypothesis during the decoding process. Suppose we have already calculated DBiCS of a partial translation hypothesis $\mathbf{e}'$, and $\mathbf{e}'$ is being expanded with a phrase pair $\langle f_{i_1}^{i_2}, e_{j_1}^{j_2}\rangle$ to be a new partial translation hypothesis. Firstly, we can generate the DBiCS of $\langle f_{i_1}^{i_2}, e_{j_1}^{j_2}\rangle$. Then the DBiCS of the new translation hypothesis is obtained by extending the preceding DBiCS of $\mathbf{e}'$ with the DBiCS of $\langle f_{i_1}^{i_2}, e_{j_1}^{j_2}\rangle$. Consequently, the DBiCSNNLM scores of the new translation hypothesis can be accumulated by Eq.(8).

Because the above decoding process involves in frequent compuatation of the normalization term, $Z(U_{i-1}, U_{i-2}, U_{i-3}; \theta)$, the naive implementaion makes the decoding very inefficient. To address this issue, we employ the technique of self-normalization, which forces the normalization term to be close to one during the training (see next Subsection), following (Vaswani et al. 2013; Devlin et al. 2014). As a result, one can ignore $Z(U_{i-1}, U_{i-2}, U_{i-3}; \theta)$ in Eq.(8) to speedup the decoding.

4.2 DBiCSNNLM Training

Although the proposed DBiCSNNLM consists of BiCNN and FFNN, they are not isolated from each other and thus being optimized their parameters $\theta = \langle \theta_1, \theta_2 \rangle$ jointly, where θ_1 is the parameters of FFNN and θ_2 is parameters of BiCNN.

Given aligned bilingual corpus with source-dependency trees, we can obtain many DBiCSs, each of which is corresponding to one bilingual sentence pair. Based on these DBiCSs, we can collect a set of 4-DBiCU tuples, which is denoted as $\mathcal{U}=\{\langle U_1^i, U_2^i, U_3^i, U_4^i\rangle | i=1, \cdots, N\}$. Formally, we maximize the regularized log-likelihood on Eq.(8), with the self-normalization term as its regularization:

$$\ell(\mathcal{U};\theta) = \sum_{i=1}^{N} \big(\log P(U_i \mid U_{i-1}, U_{i-2}, U_{i-3}; \theta) - \alpha \log^2 Z(U_{i-1}, U_{i-2}, U_{i-3}; \theta) \big) \quad (10)$$

where α is the regularizer and it is set to be 0.1 as the (Devlin et al. 2014).

However, since a DBiCU U is a tuple of words, it is inefficient to exactly calculate $Z(U_{i-1}, U_{i-2}, U_{i-3}; \theta)$. Instead, inspired by (Vaswani et al. 2013), we employ the noise contrastive estimation to approximate it:

$$Z(U_{i-1}, U_{i-2}, U_{i-3}; \theta) \approx \sum_{\bar{U} \in NB(U_i)} \exp\big(\phi(\bar{V}, V_{i-1}, V_{i-2}, V_{i-3}; \theta_2)\big) \quad (11)$$

where $NB(U_i)$ denotes the neighborhood of a gold DBiCU, i.e. $U_i = \langle M_{\mathbf{e}}^i \| PA, SI, M_{\mathbf{f}}^i, CH \rangle$. Observing that the alignment A and source-dependency tree $T_{\mathbf{f}}$ are fixed once the bilingual corpus is given, we specify the $NB(U_i)$ as the set of DBiCUs: each DBiCU is with form of $\langle M_{\mathbf{e}}' \| PA, SI, M_{\mathbf{f}}^i, CH \rangle$, satisfying $|M_{\mathbf{e}}'| = |M_{\mathbf{e}}^i|$; and it is generated by the IBM Model 1 distribution (Brown et al. 1993), inspired by (Cherry 2016).

We employ the stochastic gradient descent as the optimization algorithm, and the gradient of loss is calculated via the standard backward propagation (Bengio et al. 2003).

5 Experiments

5.1 Data Settings

We built a phrase-based Chinese-to-English SMT system by using Moses (Koehn et al. 2007), and contains a hierarchical reordering model (Galley and Manning 2008) and

a 5-gram LM trained on the Xinhua portion of Gigaword corpus using the srilm toolkit [3]. The training data contains 1.46 million sentence pairs from the LDC dataset[4]. The stanford dependency parser (Chang et al. 2009) was used to generate the dependency tree of Chinese. Word alignments were generated with the GIZA++ toolkit [5]. The Minimum error rate training (MERT) (Och 2003) was used to optimize the feature weights on the NIST02 test set, and test on the NIST03/NIST04/NIST05 test set. Translation quality is evaluated by case-insensitive BLEU-4 metric (Papineni et al. 2002). Paired bootstrap sampling (Koehn 2004) was performed to test the significance in BLEU score differences. The reported results were averaged over three MERT runs.

Most models had a vocabulary size of 50k. We used word2vec toolkit [6] to generate each word (100 dimensions) for historical DBiCUs, and each word (500 dimensions) for predicted DBiCU. These parameters were optimized by 10 epochs of stochastic gradient descent, using minibatch size 500 and a learning rate of 1.

5.2 Effect of DBiCS

System	MT03	MT04	MT05	Avg
baseline	34.59	35.41	33.12	34.37
+BiLM	35.11†	35.79†	33.56†	34.80†
+OSM	35.24†	36.05†	33.83†	35.04†
+DBiLM	35.31†	35.75†	33.80†	34.95†
+DBiCSLM	35.53†*	36.17†*	34.14†*	35.28†*

Table 1: Chinese-English NIST Results. "+" stands for adding the corresponding model to the baseline system. **AVG** = average BLEU scores for test sets. "†" means that the model significantly outperforms the baseline systems with $p < 0.05$. "*" indicates that the model is significantly better than +BiLM, +OSM and +DBiLM with $p < 0.05$.

In order to test the effectiveness of DBiCS, we learn a 5-gram DBiCS language model (DBiCSLM) using SRILM toolkit, thus being in contrast with the 5-gram bilingual language model (BiLM) (Niehues et al. 2011), 5-gram operation sequence model (OSM) (Durrani et al. 2013) and 5-gram Dependency-Based BiLM (DBiLM) (Garmash and Monz 2014) trained using the SRILM Toolkit.

In Table 1, it is observed that the DBiCSLM is +0.9 points higher than the baseline, which shows that the proposed DBiCS can improve the performance of machine translation. Moreover, the performance improves upon the BiLM, OSM and DBiLM by +0.48, 0.24 and +0.31 BLEU points in the same settings, respectively. These results further verify the effectiveness of the proposed DBiCS.

5.3 Effect of DBiCSNNLM

To have a comprehensive view on the capacity of DBiCSNNLM, we compare it with two neural network models.

[3]http://www.speech.sri.com/projects/srilm/download.html

[4]LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

[5]http://www.statmt.org/moses/giza/GIZA++.html

[6]https://code.google.com/archive/p/word2vec/

System	MT03	MT04	MT05	Avg
baseline	34.59	35.41	33.12	34.37
+NNJM	35.74†	36.79†	34.29†	35.60†
+DBiCSFFNN	35.56†	36.61†	33.92†	35.36†
+DBiCSNNLM	36.43†*	37.57†*	34.84†*	36.28†*

Table 2: Chinese-English NIST Results. "†" means that the model significantly outperforms the baseline systems with $p < 0.05$. "*" means that the model is significantly better than +NNJM and +DBiCSFFNN with $p < 0.05$.

The first one is the well-known NNJM (Devlin et al., 2014), and the other is called DBiCSFFNN, which is a feed forward neural network but treating each DBiCU instead of word as a token. In Table 2, it is observed that the proposed DBiCSNNLM is +1.9 points higher than the baseline, which shows that our model can effectively represent context with source-side dependency information for improving machine translation. Then the DBiCSNNLM gains +0.68 and +0.92 BLEU points over the NNJM and DBiCSFFNN respectively. These results indicate that out neural network architecture can more effectively encode context information for better translations than NNJM and DBiCSFFNN.

System	MT03	MT04	MT05	Avg
baseline+NNLM	35.13	36.22	33.58	34.97
+NNJM	35.82†	36.56†	34.5†	35.63†
+DBiCSFFNN	35.74†	36.86†	34.16†	35.59†
+DBiCSNNLM	36.76†*	37.97†*	35.21†*	36.64†*

Table 3: Chinese-English NIST Results with a 5-gram neural network language model (NNLM) (Vaswani et al. 2013) over the target words of training data. "†" means that the model significantly outperforms the baseline+NNLM systems with $p < 0.05$. "*" means that the model is significantly better than +NNJM and +DBiCSFFNN with $p < 0.05$.

To further verify the generality of the proposed DBiCSNNLM, we implement it on top of another neural system, baseline+NNLM, which is augmented with a NNLM over the baseline. In Table 3, we find that +DBiCSNNLM achieves significant improvements baseline+NNLM, and it gains over both +NNJM and +DBiCSFFNN as expected. In addition, we find that baseline+NNLM+NNJM is comparable to baseline+NNJM as depicted in Table 2. This shows that the gains of NNLM are absorbed into those of NNJM. On the other hand, DBiCSNNLM is orthogonal to NNLM, because baseline+NNLM+DBiCSNNLM still gains over baseline+DBiCSNNLM.

5.4 Effect on K-best Rescoring

We also apply the presented DBiCSNNLM to rescore the 1000-best translation results produced by the baseline and make a comparison to Structured Output Layer (SOUL) (Le, Allauzen, and Yvon 2012) model, NNJM (Devlin et al. 2014), jointly translation and reordering model with feed-forward neural network (JTRFFNN) (Guta et al. 2015), and minimum translation modeling with recurrent neural network (MTURNN) (Hu et al. 2014).

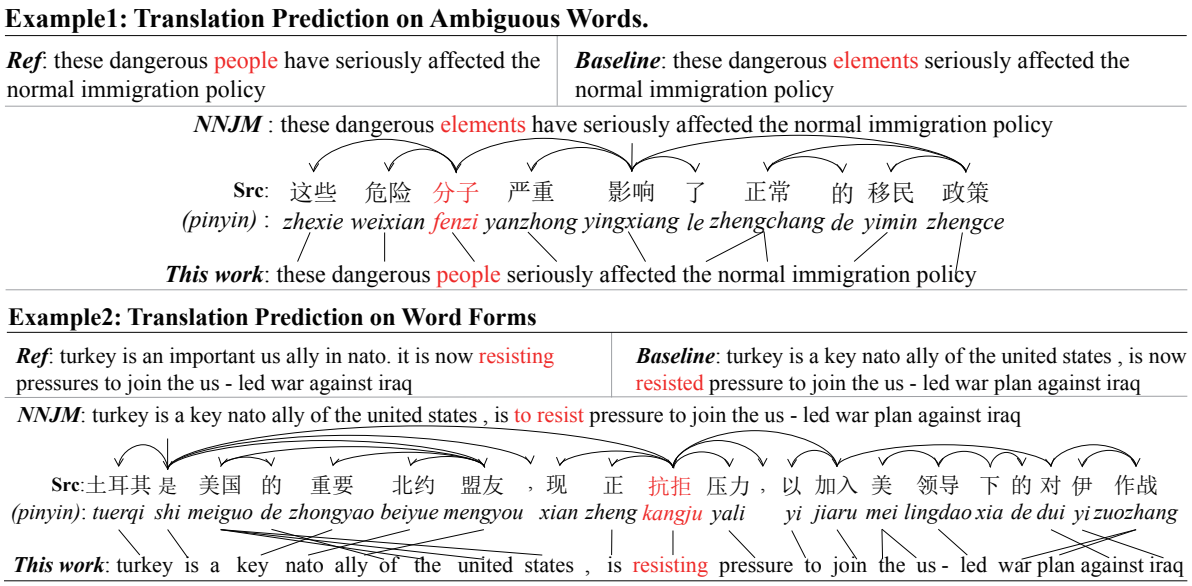

Figure 5: Output Sample Sentences.

System	MT03	MT04	MT05	Avg
baseline(Dec)	34.59	35.41	33.12	34.37
SOUL	34.73†	35.96†	33.42†	34.70†
NNJM	35.02†	36.10†	33.72†	34.94†
JTRFFNN	34.81†	35.76†	33.26†	34.60†
MTURNN	35.10†	36.16†	33.89†	35.05†
DBiCSNNLM	35.21†*	36.40†*	33.77†	35.12†*

Table 4: Comparison of the proposed DBiCSNNLM in 1000-best rescoring results with other models. "†" means that the model significantly outperforms the baseline systems with $p < 0.05$. "*" indicates that the model is significantly better than all the Comparison systems with $p < 0.05$.

In the Table 4, it is observed that the proposed DBiCSNNLM performs well when used for rescoring the baseline results (+0.69), and the gain is on average better than SOUL (+0.36), NNJM (+0.12), JTRFFNN (+0.3) and MTURNN (+0.11). These results mean that the DBiCSNNLM has a greater distinguishing ability to select better translation from 1000-best lists. Meanwhile, it is observed that the corresponding results in Table 4 are on average inferior to that of Table 2. It indicates that our method provides more translation information in the decoding than in the rescoring.

5.5 Sample Analysis

In Example 1 of Figure 5, we compare our model with both baseline and NNJM on the translation of an ambiguous word. By using BiCNN to encode the long-distance dependencies "*yingxiang, yanzhong le zhengce, fenzi, zhexie weixian*" as a context, our model can correctly translate "*fenzi*" into "people" instead of "elements" as both baseline and NNJM do. Note that the last word "*zhengce*" is very informative for the correct translation, but it is far away from "*fenzi*" such that it is not easy to be focused on by NNJM with a window-based FFNN.

In addition, the Example 2 of Figure 5 shows that our model can even distinguish those words with the same meaning but different word forms while both baseline and NNJM fail. To figure out the details, we calculate the DBiCSNNLM scores in Eq.(8) for three candidate DBiCUs corresponding to "resisting", "resisted" and "to resist" during the decoding process. Then we found that the model score (log) of "resisting" is -3.64, while those of "resisted" and "to resist" are -4.67 and -4.03. As a result, our model can correctly translate the "*kangju*" into "resisting" instead of "resisted" and "to resist", since the feature weight of DBiCSNNLM is positive after tuning. The main reason why our model can translate the accurate word form is that it encodes a dependency structure with neural networks. Therefore, the learned neural model can distinguish the different translations of word form , which is in line with the functional similarity findings in (Levy and Goldberg 2014).

Therefore, these examples realize our intuition: encoding a dependency structure with neural networks is able to capture not only the long-distance dependencies for ambiguous words translation but also functional similarity for word forms translation.

6 Conclusion and Future Work

In this paper, we proposed a novel approach to encoding the source-side long-distance dependency information for translation prediction. The proposed DBiCSNNLM can dynamically represent contexts for translation prediction at different time steps. By explicitly encoding the dependency structure, our model not only encodes the long-distance dependencies but also captures the functional information for better translation prediction on both ambiguous words and word forms. The experiments showed that the DBiCSNNLM can significantly improve translation performance over a strong base-

line, and verify the effectiveness of structure clues in the context. In the future, we will explore richer syntactic or semantic information from the context to improve translation.

7 Acknowledgments

We would like to thank three anonymous reviewers for many invaluable comments and suggestions to improve our paper. This work is supported by National Natural Science Foundation of China (NSFC, 91520204), and the State Key Development Program for Basic Research of China (863 Program, 2015AA015405).

References

Auli, M.; Galley, M.; Quirk, C.; and Zweig, G. 2013. Joint language and translation modeling with recurrent neural networks. In *Proceedings of EMNLP2013*, 1044–1054.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR2015*.

Bengio, Y.; Ducharme, R.; Vincent, P.; and Janvin, C. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.* 3:1137–1155.

Brown, P.; Della Pietra, V.; Della Pietra, S.; and Mercer, R. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics* 263–311.

Carpuat, M., and Wu, D. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of EMNLP-CoNLL2007*, 61–72.

Chan, Y. S.; Ng, H. T.; and Chiang, D. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of ACL2007*, 33–40.

Chang, P.-C.; Tseng, H.; Jurafsky, D.; and Manning, C. D. 2009. Discriminative reordering with chinese grammatical relations features. In *Proceedings of NAACL2009*, 51–59.

Cherry, C. 2016. An empirical evaluation of noise contrastive estimation for the neural network joint model of translation. In *Proceedings of NAACL2016*, 41–46.

Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research.* 2493–2537.

Crego, J. M., and Yvon, F. 2010. Improving reordering with linguistically informed bilingual n-grams. In *Proceedings of COLING2010*, 197–205.

Devlin, J.; Zbib, R.; Huang, Z.; Lamar, T.; Schwartz, R.; and Makhoul, J. 2014. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of ACL2014*, 1370–1380.

Durrani, N.; Fraser, A.; Schmid, H.; Hoang, H.; and Koehn, P. 2013. Can markov models over minimal translation units help phrase-based smt? In *Proceedings of ACL2013*, 399–405.

Durrani, N.; Schmid, H.; and Fraser, A. 2011. A joint sequence translation model with integrated reordering. In *Proceedings of ACL2011*, 1045–1054.

Eriguchi, A.; Hashimoto, K.; and Tsuruoka, Y. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of ACL2016*, 823–833.

Galley, M., and Manning, C. D. 2008. A simple and effective hierarchical phrase reordering model. In *Proceedings of EMNLP2008*, 848–856.

Garmash, E., and Monz, C. 2014. Dependency-based bilingual language models for reordering in statistical machine translation. In *Proceedings of EMNLP2014*, 1689–1700.

Guta, A.; Alkhouli, T.; Peter, J.-T.; Wuebker, J.; and Ney, H. 2015. A comparison between count and neural network models based on joint translation and reordering sequences. In *Proceedings of EMNLP2015*, 1401–1411.

Hu, Y.; Auli, M.; Gao, Q.; and Gao, J. 2014. Minimum translation modeling with recurrent neural networks. In *Proceedings of EACL2014*, 20–29.

Koehn, P.; Hoang, H.; Birch, A.; Callison-Burch, C.; Federico, M.; Bertoldi, N.; Cowan, B.; Shen, W.; Moran, C.; Zens, R.; Dyer, C.; Bojar, O.; Constantin, A.; and Herbst, E. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL2007*, 177–180.

Koehn, P. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP2004*, 388C995.

Le, H.-S.; Allauzen, A.; and Yvon, F. 2012. Continuous space translation models with neural networks. In *Proceedings of NAACL2012*, 39–48.

Levy, O., and Goldberg, Y. 2014. Dependency-based word embeddings. In *Proceedings of ACL2014*, 302–308.

Liu, l.; Watanabe, T.; Sumita, E.; and Zhao, T. 2013. Additive neural networks for statistical machine translation. In *Proceedings of ACL2013*, 791–801.

Liu, L.; Utiyama, M.; Finch, A. M.; and Sumita, E. 2016. Neural machine translation with supervised attention. In *Proceedings of COLING2016*.

Luong, T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP2015*, 1412–1421.

Meng, F.; Lu, Z.; Wang, M.; Li, H.; Jiang, W.; and Liu, Q. 2015. Encoding source language with convolutional neural network for machine translation. In *Proceedings of ACL-IJCNLP2015*, 20–30.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS2013*, 3111–3119.

Mikolov, T. 2012. *Statistical Language Models based on Neural Networks*. Ph.D. Dissertation, Brno University of Technology.

Niehues, J.; Herrmann, T.; Vogel, S.; and Waibel, A. 2011. Wider context by using bilingual language models in machine translation. In *Proceedings of WMT2011*, 198–206.

Och, F. J. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL2003*, 160–167.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL2002*, 311–318.

Shen, L.; Xu, J.; and Weischedel, R. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL2008*, 577–585.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of ACL2015*, 1556–1566.

Vaswani, A.; Zhao, Y.; Fossum, V.; and Chiang, D. 2013. Decoding with large-scale neural language models improves translation. In *Proceedings of EMNLP2013*, 1387–1392.

Zhang, J.; Zhang, D.; and Hao, J. 2015. Local translation prediction with global sentence representation. In *Proceedings of IJCAI 2015*, 1398–1404.