Syntax-Based Context Representation for Statistical Machine Translation

Kehai CHEN¹, Student Member, Tiejun ZHAO¹†, Nonmember, and Muyun YANG¹, Member

SUMMARY Learning semantic representation for translation context is beneficial to statistical machine translation (SMT). Previous efforts have focused on implicitly encoding syntactic and semantic knowledge in translation context by neural networks, which are weak in capturing explicit structural syntax information. In this paper, we propose a new neural network with a tree-based convolutional architecture to explicitly learn structural syntax information in translation context, thus improving translation prediction. Specifically, we first convert parallel sentences with source parse trees into syntax-based linear sequences based on a minimum syntax subtree algorithm, and then define a tree-based convolutional network over the linear sequences to learn syntax-based context representation and translation prediction jointly. To verify the effectiveness, the proposed model is integrated into phrase-based SMT. Experiments on large-scale Chinese-to-English and German-to-English translation tasks show that the proposed approach can achieve a substantial and significant improvement over several baseline systems.

key words: syntax context representation, tree-based neural network, translation prediction, statistical machine translation

1. Introduction

Context representation for translation prediction has attracted much attention in statistical machine translation (SMT), and in particular, there are many studies involving neural networks (NNs) for context representation. For example, some work [1]–[6] uses NNs to represent context as sequential source or target words when predicting a target word. One of the advantages is that it uses NNs to model context, including higher order target n-grams [1], [2], target n-grams with their “relevant” source words [3], [4], or even with an entire source sentence [5], [6]. Despite their success, these methods focused on sequential context without taking syntactic structure information into consideration.

Meanwhile, context with syntactic information has been well demonstrated to be effective for improving the SMT performance. For instance, linguistic constraints from source syntax trees have been used as features in a linear classifier to learn phrase boundaries [7]–[9], and source part-of-speech tagging information has been used to improve translation prediction [10]–[12]. The syntactic features extracted by these methods are not sparse, and hence they can be integrated directly into linear models to improve translation performance [13]. However, for richer syntactic information in context, such as linguistic syntax subtrees, there is a severe data sparsity problem for a linear model [13], and hence the linguistic syntax subtrees have not often been considered as potential models for translation context.

In this article, we propose a new neural network with a tree-based convolutional architecture, which is able to explicitly learn structural syntax information in translation context and thus improve translation prediction. Specifically, we first detect minimum syntax subtrees from a source parse tree, and then convert parallel sentence pairs with source parse trees into syntax-based linear sequences. On the basis of a syntax-based linear sequence, the proposed tree-based convolutional network is used to learn syntax-based context representation and translation prediction jointly. In particular, the tree-based network has a certain capability of vector composition to greatly alleviate the data sparsity caused by relatively large syntactic units, and it enables to compute a dynamically syntactic context representation for translation prediction at each time step.

This article makes the following contributions:

• A new syntax subtree unit is defined to identify source linguistic syntax units, and these units are projected into an equivalent-meaning target sentence by word alignments, thereby generating a syntax-based linear sequence. The sequence can encode both source-side syntactic structures and target-side ordered constraints.

• The proposed model consists of a feedforward neural network with a tree-based convolutional neural network. Specifically, the structural NN can encode contextual syntax in a structural manner rather than in a sequential manner, and it can learn representation for each syntax-based linear sequence via compositional representation over a relatively small word and syntax tokens, thereby greatly alleviating data sparsity.

• During the decoding processing, the proposed model can encode context with syntactic structures for translation prediction dynamically in different time steps. Experimental results on Chinese-to-English and German-to-English translation tasks show that the proposed method achieves a substantial and significant improvement over several baseline systems.
2. Statistical Machine Translation

SMT aims to translate a source language sentence \( f = \{ f_1, \ldots, f_I \} \) into an equivalent-meaning target language sentence \( e = \{ e_1, \ldots, e_J \} \), where \( I \) and \( J \) are the lengths of the source and target sentences, respectively. According to the method of Brown et al. [14], the translation processing is represented formally as the Eq.(1):

\[
\argmax_x P(e|f) = \argmax_e P(f|e)P(e),
\]

where \( P(f|e) \) denotes a separate translation model which is used to ensure that the source meanings are mapped into the target translation as adequately as possible, and \( P(e) \) denotes a language model which is used to ensure that the target translation is understood by the target language speaker as fluently as possible. Generally, the language model used a language model which is used to ensure that the target sentence, where \( P(s) \) monolingual context with the source-side local context

\[
P(e) = \prod_{i=1}^{n} p(e_i|c_i).
\]

where \( p(e_i|c_i) \) denotes the conditional probability of the current target word \( e_i \) given target historical context \( c_i = [e_{i-n+1}, \ldots, e_{i-1}] \):

\[
p(e_i|c_i) = p(e_i|e_{i-n+1}, \ldots, e_{i-1}),
\]

where \( n \) is the order of language model.

Most notably, Devlin et al. [4] extended the target monolingual context with the source-side local context

\[
s_i = s_{a_i-G}^{a_i+G},
\]

where \( s_{a_i-G}^{a_i+G} \) denotes source words centering around the source position \( a_i \) aligned by the current target word \( e_i \), and source local context window is \( 2G + 1 \).

However, just as we discussed in the Introduction section that the source structural syntax information, especially source syntax subtrees, plays an indispensable role in accurate translation prediction. Compared with the word-level context in language model, source structural syntax information has shown greatly success for translation model, which focused on improve the adequate of target translations. In other words, the source structural syntax information cannot be adequately used to improve the fluency of target translations. In this paper, we will explore source syntax structural information for improving the fluency of target translations.

3. Syntax-Based Linear Sequence

3.1 Minimum Syntactic Subtree

Inspired by work on learning translation rules for syntax-based SMT [15–17], we define a new minimum syntax subtree (MSST) to identify source-side syntactic units.

![Fig. 1](http://example.com/fig1.png)

Given a source syntax tree \( T_f \), the definition of an MSST \( t \) is as follows:

- **Condition 1**: There are no greater than seven leaf nodes contained in \( t \), because the maximum length of a translation unit (source or target word) is set to seven in the phrase-based translation model [18].
- **Condition 2**: The depth of each \( t \) is no greater than four, thereby enabling the source syntax tree \( T_f \) to be converted into a proper size sequence of \( t \).
- **Condition 3**: All leaf nodes of \( t \) are terminal words. In this work, these terminal words from \( t \) are regarded as a linguistic phrase.

The MSST \( t \) is as shown in the dashed boxes in Fig. 1.

The motivation for introducing MSST is that the subtrees often used in tree-based SMT are not used to project syntax structures between bilingual parallel sentence pair directly, because there are two kinds of syntax subtrees in the tree-based SMT: subtrees with nonterminal symbols and subtrees with a terminal symbol. Specifically, a subtree with nonterminal symbols can lead to an uncertain sequence of syntax subtrees. In other words, during translation decoding, a nonterminal can be expanded to a subtree or many subtrees, leading to multiple sequences of syntax subtrees for one sentence instead of a determinate and single subtree sequence. Meanwhile, a subtree with a terminal symbol degrades bilingual syntactic projection into traditional bilingual word projection, with the result that the phrase and syntactic structure cannot be captured adequately.

In contrast, the proposed MSST aims to overcome the constraints of subtree units in tree-based SMT. First, **Condition 1** and **Condition 2** guarantee that translation units with source syntactic structures can match the existing lexicalization translation units as closely as possible. Second, **Condition 3** says that the learned MSST does not consist of the nonterminal symbol to generate a unique syntax subtree sequence.
In summary, MSSTs will be used to model the shared syntactic structures between bilingual parallel sentence pairs, thereby capturing context with syntactic information.

### 3.2 Generation of Syntax-Based Linear Sequences

Because MSST is shared between bilingual parallel sentence pairs, we use it to generate a syntax-based linear sequence (SLS) from a bilingual parallel sentence pair with a source syntax tree. An SLS is defined by a sequence of syntax-based linear sequence units (SLSUs). Specifically, we use word alignments to project each MSST into a consecutive sequence of target words. Because MSSTs are determined given bilingual parallel sentence pairs with source syntax trees and word alignments, we call the pair of the source MSST and corresponding target words an SLSU.

Formally, the proposed SLS is defined as a sequence \( S \) that is a new conversion of a given source language sentence \( f \) with source syntax tree \( T_f \), target language sentence \( e \) and corresponding word alignment \( a \). First, these MSSTs are identified from \((f, T_f)\). Second, these source MSSTs are projected to the target language by using word alignments, and these shared source POS tags are regarded as root nodes of bilingual word pairs, as shown in the second subtree in Fig. 2. Third, each SLSU is converted into a binary subtree for simplifying SLSU, as shown the third subtree in Fig. 2. Finally, the SLS \( S \) is implemented by left-to-right serialization of \((f, e, T_f, a)\) depending on MSSTs and word alignments \( a \), as shown in Fig. 3 as \(\{U_1, U_2, U_3, U_4\}\). In the process of serialization, the source language part of the SLS can be consecutive or discontinuous, but target language of SLS must be consecutive. Consequently, the SLS \( S \) is denoted by the Eq.(5):

\[
S = \{U_1, \cdots, U_m, \cdots, U_M\},
\]

where \( M \) is the number of SLSUs in the SLS.

### 3.3 Issues: Target Discontinuity and Unaligned Words

Two issues that we have ignored so far are the handling of an MSST projection with discontinuous targets, and the handling of an MSST projection with unaligned target words. The SLS is generated linearly in left-to-right order. This assumption becomes problematic as a result of target-side discontinuities (See Fig. 4 (a)). \( MSST_1 \) cannot be generated because of the intervening projections “c→CD”. Hence, in accordance with the post-processing heuristic [19], we modify the word alignments to remove such cases. Specifically, when a source word is aligned to discontinuous targets, the first link to the least frequent target word is identified, and the group of links containing this word is retained while the others are deleted, as shown in Fig. 4 (a).

The second problem is unaligned target-side projections\(^1\), such as “\(d \rightarrow \epsilon\)” in Fig. 4 (b). Insertion of target-side “\(\epsilon\)” tokens during decoding is a serious problem, because there is no evidence regarding when to hypothesize such words. These cases are dealt with by merging such projections to the MSST on the left or right based on lexical probabilities obtained from IBM Model 1 [20]. Notice

\(^1\)The “spurious” indicates that the source word is aligned to “NULL” of target sentence.
Fig. 5  Tree-based convolutional neural network. In the input layer, each node of the syntax linear unit is firstly mapped to a 100-dimension vector representation; we then extract a structured feature information for each node through a tree-based convolution operation; thirdly, we use the proposed heuristic pooling to generate a 3-slot structural vector representation; finally, slot-wise averaging is performed over the feature outputs from heuristic pooling and obtains a vector representation of a syntax linear unit.

4. Representation Learning for SLSU

This section introduces the proposed tree-based convolutional neural network (TCNN), which is a variant of the standard convolutional neural network (CNN) [21], to learn a compact representation for each SLSU, thereby encoding context with syntax information. Figure 5 shows the tree-based convolutional architecture for learning vector representation for an SLSU. TCNN consists of an input layer, a tree-based convolutional layer, pooling heuristic layer, and an output layer.

4.1 Input Layer

The input layer is a subtree of SLSU, in which are two kinds of nodes, where leaf nodes are words and non-leaf nodes represent grammatical constituents, e.g., VP, ADVV. Each node in an SLSU is represented as a fixed-size, real-valued vector $v \in \mathbb{R}^d$, where $d$ is dimensional of vector. For example, there are $H$ nodes in a SLSU $U_m$.

4.2 Tree-Based Convolution

We now consider tree-based convolution processing in TCNN. Specifically, the current processed node is $h$, its left child $h_l$, and its right child $h_r$. Their vector representations are $v_h, v_l$, and $v_r$. The convolution operation is shown as Eq.(6):

$$v'_h = \tanh(W_h v_h + W_{hl} v_{hl} + W_{hr} v_{hr} + b),$$

where tanh is the activation function, and $W_h, W_{hl},$ and $W_{hr}$ are weight matrices, and $b$ is the bias term. Note that $v_{hl}$ or $v_{hr}$ are padded to be zero when left or right children of a node do not exist. After tree-based convolution, structural features in an SLSU are extracted, thereby generating a new tree, for example, $\{v'_1, v'_2, v'_3, v'_4, v'_5, v'_6, v'_7, v'_8, v'_9, v'_{10}\}$ in Fig. 5. The new tree has exactly the same shape and number of nodes as the original one, which varies among different SLSUs.

Tree-based convolution windows can be extended straightforwardly to arbitrary depths. The complexity is exponential in the depth of the window, but linear in the number of nodes. Hence, in contrast to “flat” CNNs, tree-based convolution does not add to computational cost, providing the structural information to process at a particular decoding time step. In our experiments, the depth of the convolution windows is set to two.
4.3 Pooling Heuristic

To preserve more information over different parts of the SLSU, we take a common dynamic pooling [22]. Specifically, 3-slot pooling is used for max-pooling, as shown in Fig. 5. The design criteria for our pooling heuristic include the following:

- Nodes that are pooled to one slot are to “neighboring” from some viewpoint.
- Each slot is to similar numbers of nodes as many as possible, that expected to be pooled to it.

Thus, equal amounts of information are aggregated along different parts of the tree structure. More specifically, if an SLSU has a maximum depth D, nodes of less than γ-D layers are pooled to a TOP slot (γ is set to 0.6), and lower nodes are pooled to slots Left or Right according to their relative positions with respect to the root node. A node-wise max operation is performed over all nodes of Top, Left, and Right, respectively. Finally, there are three d-dimensions feature outputs: v_T for over TOP slot, v_L for over Left slot, and v_R for over Right slot. For example, Fig. 5 show a Pooling Heuristic. The γ-D is equal to 0.6×4=2.4≈2. The v_T, v_L, and v_R are computed by the Eq.(7):

\[
\begin{align*}
v_T &= \max(v'_1, v'_2, v'_7), \\
v_L &= \max(v'_3, v'_4, v'_5, v'_6), \\
v_R &= \max(v'_8, v'_9, v'_10).
\end{align*}
\]

4.4 Output Layer

The output layer is a fully connected layer and is chosen according to specific tasks; for example, for binary classification tasks [23], this layer is logistic regression. In this paper, slots-wise averaging is performed over the feature outputs v_T, v_L, and v_R of pooling heuristic layer, whereby the vector representation v_Um of the SLSU is obtained according to the Eq.8:

\[
v_{um} = \text{average}(v_T + v_L + v_R).
\]

Therefore, the above TCNN plays the role of function f parameterized by \(\theta_1\), which maps an SLSU \(U\) into \(v_{um}\):

\[
v_{um} = f(U_m; \theta_1).
\]

Note that \(v_{um}\) in Eq. (8) is a vector rather than a scalar and will be transformed into a scalar to score a translation hypothesis later. There are many SLSUs in an SLS S. Hence, the SLS is a sequence of these ordering \(v_{um}\):

\[
V_S = [v_{u_1}, \ldots, v_{u_m}, \ldots, v_{u_M}],
\]

where S is the SLS as in Sect. 3, \(U_m\) is an SLSU, and \(M\) is the number of SLSU in the S.

5. Translation Prediction with Syntax Context Representation

5.1 Syntax-Based Neural Network Joint Model

With a trained TCNN, each SLSU is represented as a vector representation. In this section, a unique FFNN with the TCNN is designed to predict translation using syntactic context representation over the SLS. The proposed NN architecture as illustrated in Fig. 6, consists primarily of two components. First, an inlayer TCNN, as described in the last section, is used to learn syntactic context representation over the SLS, and second, an outlayer FFNN is used to predict the next SLSU depending on the syntactic context representation that encodes the previous SLSUs. Since the TCNN part learns vector representations for an SLSU and the FFNN part is factorized over 4 SLSUs similar to an neural network joint model, the model is called the syntax-based neural network joint model, SNNJM.

Given a source language f and its syntax tree \(T_f\), for any translation e with alignments α, we can obtain its corresponding SLS denoted as \(\{U_1, \ldots, U_m, \ldots, U_M\}\). Then we define the following model to score \(\langle f, e, T_f, a \rangle\):

\[
P(f, e, T_f, a) = \prod_{m=1}^{M} P(U_m | U_{m-1}, U_{m-2}, U_{m-3}; \theta_1)
\]

\[
= \prod_{m=1}^{M} \frac{\exp(\phi(U_m, U_{m-1}, U_{m-2}, U_{m-3}; \theta_2))}{Z(U_{m-1}, U_{m-2}, U_{m-3}; \theta)},
\]

where \(Z(U_{m-1}, U_{m-2}, U_{m-2}; \theta)\) is used as the normalization:

\[
Z(U_{m-1}, U_{m-2}, U_{m-2}; \theta) = \sum_{U_m} \exp(\phi(U_m, U_{m-1}, U_{m-2}, U_{m-3}; \theta_2)),
\]

where \(\phi\) is an FFNN parameterized by \(\theta_2\); and \(\theta=(\theta_1, \theta_2)\) denotes all of the model parameters including both the TCNN and the FFNN, and \(M\) is the number of SLSU in a SLS S.

Since Eq.(11) is factorized over the SLSUs, calculating the SNNJM scores of a partial translation hypothesis incrementally during the decoding process is straightforward. Suppose that we have already calculated the SLS of a partial translation hypothesis \(e'\), and that \(e'\) is expanded with
a phrase pair $\langle f_{j_i}^{\alpha}, e_{i}^{\alpha} \rangle$ with source syntax subtree $T_f$ and word alignment $\alpha$ to be a new partial translation hypothesis. First, we can generate the SLS of $\langle f_{j_i}^{\alpha}, e_{i}^{\alpha}, T_f, \alpha \rangle$. Then we can obtain the SLS of the new translation hypothesis by extending the preceding historical SLS of $\epsilon'$ with the SLS of $\langle f_{j_i}^{\alpha}, e_{i}^{\alpha}, T_f, \alpha \rangle$. Consequently, the SNNJM scores of the new translation hypothesis can be accumulated according to Eq.(11).

During the decoding processing, we build a syntax linear unit for each phrasal translation rule according to the minimum syntactic subtree constraints. If the phrasal translation rule is not a complete subtree, we build a pseudo subtree in which these locations without words will be padded by using a specific placeholder.

5.2 SNNJM Training

Although the proposed SNNJM consists of an FFNN and a TCNN, they are not isolated from each other, and thus both of their parameters $\theta = (\theta_1, \theta_2)$ are optimized jointly, where $\theta_1$ is the parameter set of FFNN and $\theta_2$ is the parameter set of TCNN.

Given an aligned bilingual corpus with source syntax trees and word alignments, we can obtain many SLSs, each of which corresponds to one bilingual sentence pair. Based on these SLSs, we can collect a set of 4-gram SLSUs, and denote as $T=\langle U^k_1; U^k_2; U^k_3; U^k_4 \rangle[k=1,2,\ldots,K]$. Formally, we maximize the regularized log-likelihood on Eq.(11), with the self-normalization term:

$$\ell(T; \theta) \approx \sum_{k=1}^{K} \sum_{m=1}^{M} \left( \phi(U^k_m | U^k_{m-3}, U^k_{m-2}, U^k_{m-1}; \theta_2) \right)$$

$$-\alpha \cdot \log Z(U^k_{m-3}, U^k_{m-2}, U^k_{m-1}; \theta_1),$$

where $\alpha$ is the regularizer and is set to 0.1 [4]

However, since an SLSU $U$ is a bilingual phrase pair with source syntactic structure, calculating $Z(U^k_3, U^k_2, U^k_1, \theta)$ is inefficient. Instead, we use the noise contrastive estimation [24] to approximate it:

$$Z(U^k_3, U^k_2, U^k_1, \theta) \approx \sum_{\tilde{U} \in NBU(U^k_3)} \exp(\phi(\tilde{U}, U^k_3, U^k_2, U^k_1; \theta_2)).$$

where $NBU(U^k_3)$ is a neighborhood of a gold SLSU, i.e. $U^k_4=\langle f_{j_i}^{\alpha}, e_{i}^{\alpha}, T_f, \alpha \rangle$. Observing that the alignment $\alpha$ and source-dependency tree $T_f$ are fixed once the bilingual corpus is given, we specify $NBU(U^k_3)$ as the set of SLSUs: each of its members has the form of $\langle f_{j_i}^{\alpha}, e_{i}^{\alpha}, T_f, \alpha \rangle$ and it is generated by the IBM Model 1 [20] distribution of $\langle f_{j_i}^{\alpha}, e_{i}^{\alpha} \rangle$, inspired by [25].

We use stochastic gradient descent as the optimization algorithm, and the gradient of loss is calculated using the standard backward propagation [26].

5.3 Phrase-Based SMT with SNNJM

To verify the effectiveness, the proposed SNNJM is integrated into phrase-based SMT. Specifically, at the current decoding time step, we first obtain SLSUs for each translated target word or phrase in accordance with Sect. 2, and we concatenate these new SLSUs with the previously generated SLSUs to form the current SLS. If we only take translation model and our SNNJM model into consideration, the phrase-based SMT with SNNJM formally denotes as a modified version of the Eq. 1:

$$\text{argmax}_{\epsilon} P(\epsilon, f, T_f) = \text{argmax}_{\epsilon} P(\epsilon f) P(f, T_f, \alpha),$$

where $P(\epsilon, f, T_f, \alpha)$ denotes the conditional probability over the learned SLS from syntax tree $T_f$ of source sentence $f$, and according to the Eq.(2), the computing processing of $P(\epsilon, f, T_f, \alpha)$ is as the following Eq.(16):

$$P(\epsilon, f, T_f, \alpha) = \prod_{m=1}^{M} P(U_{m}\epsilon_{m}),$$

where $P(U_{m}\epsilon_{m})$ denotes the conditional probability of the current SLSU $U_{m}$ given historical SLSU context $\epsilon_{m}={U_{m+1}, \ldots, U_{m-1}}$:

$$P(U_{m}\epsilon_{m}) = P(U_{m}|U_{m+1}, \ldots, U_{m-1}).$$

where $n$ is the order of the proposed SNNJM.

6. Experiments

6.1 Data Settings

Experiments are conducted on Chinese-to-English translation tasks using a phrase-based decoder in Moses [27]. The training data include 1.46 million sentence pairs from the LDC dataset\textsuperscript{1}. The Stanford parser is used to generate the Chinese syntax tree [28]. Word alignments are generated with the GIZA++ toolkit [29]. MERT [30] is used to optimize the feature weights on the NIST02 test set, and the translation performance is evaluated on the NIST03/NIST04/NIST05/NIST06/NIST08 test sets. The case-insensitive BLEU [31] is used as the evaluation metric. The statistical significance test is performed with the pairwise re-sampling approach [32]. The reported BLEU scores are averaged over three MERT optimization runs.

The \texttt{word2vec} toolkit [1] is used to represent each word as a 100-dimensions vector\textsuperscript{2}. Most translation models have a vocabulary size of 50k. The network parameters

\textsuperscript{1}The corpus includes LDC2002E18, LDC2003E07, LDC2004E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

\textsuperscript{2}Specifically, we use the Google \texttt{word2vec} toolkit to learn the word vectors. For the training parameters, we use the continuous bag of words model; the learning rate is 0.05; the max skip length between words is 5; the number of negative examples is 10; the dimension of word vectors is 100; the number of iteration is 15; the randomly down-sampled value is 1e-4; other parameters use default configures.
are uniformly initialized between ($-0.01, 0.01$). Then the model parameters are optimized by 15 epochs of stochastic gradient descent, using minibatches of size 500 and a learning rate of $1$. We draw 100 noise samples per training example from the unigram distribution [24].

In particular, we use Nematus Toolkit\footnote{https://github.com/EdinburghNLP/nematus} to show the results of a standard attentional neural machine translation (AttNMT) [33]. We limit the source and target vocabularies to 50K, and the maximum sentence length was 80. We shuffle training set before training and the mini-batch size is 80. The word embedding dimension is 620-dimensions and the hidden layer dimension is 1000-dimensions, and the default dropout technique [34] in Nematus is used on the all layers. Our NMT models are trained about 15 EPOCHs on the training data by using ADADELTA optimizer [35], taking six days on a single Tesla P100 GPU, and the beam size for decoding was 12.

6.2 Baseline System

To obtain a comprehensive understanding regarding the capacity of our proposed model, we compare our method with several baselines. Details regarding different systems are:

- **Baseline**: a phrase-based translation model in Moses [27] contains phrasal and lexical smoothing models for both directions, word and phrase penalties, a distance-based reordering model, a hierarchical reordering model [36] and a 5-gram LM (trained on the English Gigaword corpus).
- **LM**: a language model trained on the target language of training data.
- **BiLM**: a bilingual language model [11] operating on non-monotone alignments.
- **OSM**: an operation sequence model [37] combining dependencies on bilingual word pairs and reordering information into a single framework.
- **DBiLM**: a dependency-based BiLM [12] that augments the BiLM with POS taggers of source dependency words, and add it into the Baseline.
- **NNLM**: an NN language model [24] in which an FFNN is used to predict the current target word depending on previous target words in context.
- **NNJM**: an NN joint model [4] that augments NNLM with a source context window. For example, the probability is determined with four target history words and eleven source-side local context words.
- **DNNJM**: a DBiCS-based NN joint model [38] flatten local dependency structures including parent, children, and sibling words into a word tuple, which is represented as a vector representation by a convolutional NN to model the long-distance dependency constraints. For example, the probability is determined with a source dependency context which includes n-gram dependency-based bilingual context units.
- **BCCNN**: It is similar to NNJM. In addition to the

source/target local context in a fixed-size window, the global representation of the source sentence is introduced by bilingually-constrained chunk-based convolutional NNs [5] for improving translation prediction.

- **MTURNN**: RNN-based minimum translation unit [39] models that make predictions based on an unbounded history of previous bilingual contexts. This model is used only for post-processing via k-best rescoring instead of translation decoding, because of recurrence on the target side.
- **SOUL**: structured output layer [40] model that reorders the source to match the linear order of the target, and then segments the hypothesis into minimal bilingual phrase pairs. The SOUL can be used only in k-best rescoring, because it requires long-distance reordering and a large target context.

Especially, for the order of the above models, we find that when the order is greater than five, most models achieve no further improvement, and even decline slightly. Therefore, their orders are set to 5-gram, for example, 5-gram OSM, and 5-gram NNLM. Besides, according to the difference of training method, the above models are classified into two categories: count-based methods (LM, BiLM, OSM and DBiLM) and NN-based methods (NNLM, NNJM, BCCNN, MTURNN and SOUL).

6.3 Overall Performance

Table 1 shows the translation performances on test sets measured in BLEU score. All the comparison methods, including Count-Based models and NN-Based models, outperform the Baseline system, indicating that these exist methods enhance the performance of SMT system.

1) It is clearly observed that the performance of the NN-Based models is better than that of the count-Based models. This indicates that the NN-based methods can capture syntactic and semantic information of words in context implicitly to predict translation.

2) +NNJM, +BCCNN and our +SNNJM outperform the +NNLM over the target-side monolingual context, indicating that source context information is beneficial for SMT.

3) The proposed +SNNJM is superior to +NNJM and +BCCNN. This indicates that source syntax information is useful for translation prediction. In other words, the proposed SLS can encode source syntax context to enhance translation prediction.

4) The +SNNJM gains improvements of 0.55 and 0.31 BLEU points on average over +NNJM and the +BCCNN. This indicates that our method can effectively capture the source syntax information and learn syntax-based context representation to improve the performance of SMT.

5) The Baseline gives a competitive translation performance compared with AttNMT, when given an extra large-scale target language model. The +SNNJM outperforms AttnMT model because that we used large-scale English Gigaword corpus to train an additional language model.
### Table 1
Chinese-to-English experimental results when different comparison models are integrated into translation decoding. "*" indicates that the model has statistical significance better than Baseline at p-value < 0.05 and "**" at p-value < 0.01. "†" means that the model is statistically significant difference (p-value < 0.05) from NNJM which performed the best among comparison methods and "‡" at p-value < 0.01. AVG is average BLEU scores for MT03-MT08 test sets.

| Training System | Test Sets | AVG |
|-----------------|-----------|-----|
| Baseline | NIST02 | NIST03 | NIST04 | NIST05 | NIST06 | NIST08 | AVG |
| +LM | 36.81 | 35.48 | 35.48 | 34.02 | 31.43 | 25.60 | 32.47 |
| +BiLM | 37.01 | 35.80 | 36.02 | 34.11 | 32.17 | 26.17 | 32.85 |
| +OSM | 37.25 | 36.22 | 36.72 | 34.89 | 32.24 | 26.33 | 33.28 |
| +DiBiLM | 37.27 | 36.12 | 36.65 | 34.71 | 32.11 | 26.04 | 33.12 |
| **NN-Based** | | | | | | | |
| +NNLM | 37.65 | 36.42 | 36.92 | 35.09 | 32.44 | 26.33 | 33.44 |
| +NNJN | 37.69 | 35.59 | 36.98 | 35.26 | 32.57 | 26.49 | 33.58 |
| +DNJM | 37.89 | 36.81 | 37.64 | 35.82 | 33.45 | 27.27 | 34.20 |
| +BCCNN | 37.76 | 36.76 | 37.15 | 35.51 | 32.89 | 26.81 | 33.82 |
| **SNNJM** | 37.99 | **37.11** | **37.51** | **35.68** | **33.28** | **27.09** | **34.13** |
| **SNNJM+DNJM** | 38.02 | **37.54** | **38.06** | **36.21** | **33.61** | **27.62** | **34.60** |
| +SNNJM+BCCNN | 37.67 | 35.61 | 36.06 | 33.91 | 31.67 | 25.72 | 32.59 |
| +SNNJM+KO | 37.79 | 35.75 | 36.21 | 33.93 | 31.73 | 25.61 | 32.58 |
| +SNNJM+SOUL | 37.96 | 35.89 | 33.96 | 31.37 | 25.87 | 32.59 |
| **SNNJM** | 38.02 | 36.22 | 34.34 | 31.89 | 25.98 | 32.85 |

### Table 2
Comparison of the proposed SNNJM in 100-best rescoring results with NNL, NNJM, RNNMTU and SOUL.

| System | Test Set | AVG |
|--------|----------|-----|
| Baseline | NIST03 | NIST04 | NIST05 | NIST06 | NIST08 | AVG |
| NNL | 35.26 | 35.71 | 33.70 | 31.38 | 25.34 | 32.28 |
| NNLM | 35.61 | 36.06 | 33.91 | 31.67 | 25.72 | 32.59 |
| NNJM | 35.79 | 36.16 | 34.21 | 31.99 | 25.86 | 32.90 |
| RNNMTU | 35.76 | 35.89 | 33.93 | 31.73 | 25.61 | 32.58 |
| SOUL | 35.51 | 35.97 | 33.86 | 31.37 | 25.87 | 32.59 |
| SNNJM | 35.82 | 36.22 | 34.34 | 31.89 | 25.98 | 32.85 |

6) The +SNNJM is slightly inferior to the +DNJM, but there is an improvement of BLEU scores when both of them could be integrated into the baseline together, as shown in the +SNNJM+DNJM in Table 1. This means that the information encoded in the local phrase structures and the long-distance dependency structures are complementary in translation prediction.

### 6.4 Effect on K-Best Rescoring

Table 2 show comparison results of the proposed SNNJM in 100-best rescoring results with NNL, NNJM, RNNMTU and SOUL:

1) It is observed that the SNNJM performs well when used for rescoring the baseline results (+0.47), and the gain is slightly better than for NNL (+0.36), NNJM (+0.05), RNNMTU (+0.27), and SOUL (+0.26) respectively. This indicates that the SNNJM has a greater distinguishing ability to select better translations from 100-best lists.

2) In addition, we find that the corresponding results in Table 2 are inferior, on average, to those in Table 1. This indicates that the +SNNJM provides more translation information for translation prediction in the decoding process than in post-processing rescoring.

### 6.5 Experiments on German-to-English Translation Task

To verify further the language-independence of our method, we evaluate the proposed SNNJM in a German-to-English translation task. The training data include 4.43 million bilingual parallel sentence pairs as a part of the WMT’14 dataset. The 5-gram Kneser-Ney language model is trained only on target monolingual corpora of the training data using the SRILM toolkit. MERT [30] is used to optimize the feature weights on the newstest2012 test set, and each model is tested on the newstest2013/newstest2014/newstest2015 test sets. In addition, other configurations are the same as in the Chinese-to-English translation task, for example, source language (i.e., German) syntax tree, and word alignment.

As can be seen in Table 3, the proposed +SNNJM performed better than the Baseline (+1.0 BLEU points) on average. In particular, the +SNNJM achieves improvements of 0.48 and 0.26 BLEU points on average over +NJNN and +BCCNN. These results further show that the proposed model can also effectively improve the English-to-German translation task. In other words, the proposed model is robust for improving the translation of other language pairs.

In addition, compared with experiments for Chinese-to-English task, these methods for SMT are comparable to the AttNMT. In particular, the +SNNJM achieves improvements of 0.59 BLEU points on average over AttNMT.

### 6.6 Training and Decoding Efficiency

To show training and decoding efficiency, we use a statistic of training and decoding time under the same experimental configuration for all the NN-based models. Table 4 shows model training time in the Chinese-to-English translation task, and decoding time for translating NIST03-NIST08 (6810 sentences).

1) We observe that NNJM, BCCNN, and SNNJM require more training time than NNL every epoch. This is because NNJM, BCCNN and SNNJM are based on bilingual context while NNL is based on target monolingual.
Table 3  German-to-English experimental results when different comparison models are integrated into translation decoding. “†” means that the model has a statistically significant difference (p-value < 0.05) from the NNJM, which performed the best among comparison methods. AVG is the average BLEU scores for the tst2013-tst2015 test sets. All the model is 5-gram context.

| Training | System     | Dev Set     | Test Sets     | AVG        |
|----------|------------|-------------|---------------|------------|
|          |            | newstest2012| newstest2013  | newstest2014 | newstest2015 |       |
| Baseline |            | 21.45       | 23.84         | 22.12       | 23.28       | 23.08 |
| Count-based | +BiLM       | 21.63       | 24.02         | 22.34       | 23.46       | 23.24 |
|          | +OSM        | 21.86       | 24.21         | 22.52       | 23.75       | 23.49 |
|          | +DBiLM      | 21.81       | 24.09         | 22.41       | 23.72       | 23.41 |
| NN-based | +NNLM       | 21.66       | 24.11         | 22.42       | 23.66       | 23.40 |
|          | +NNJM       | 22.20       | 24.34         | 22.69       | 24.06       | 23.70 |
|          | +DNNJM      | 22.71       | 25.12         | 23.26       | 24.74       | 24.37 |
|          | +BCCNN      | 22.26       | 24.46         | 22.71       | 24.36       | 23.84 |
|          | +NNJM+DNNJM | 22.59       | 24.88†        | 23.06       | 24.59†      | 24.18 |
|          | +SNNJM      | 22.84       | 25.49†        | 23.42†      | 24.87†      | 24.59†|
|          | +DNNJM+SNNJM| 25.80       |              |            |             |       |
|          | AttNMT      | 21.89       | 24.57         | 22.73       | 23.47       | 23.59 |

Fig. 7  Sample outputs.

Table 4  Training and decoding efficiency on the Chinese-to-English translation task.

| Models   | Training Time (Time/per epoch) | Decoding Time (Sentences/per second) |
|----------|--------------------------------|-------------------------------------|
| NNLM     | 2.49 hours                      | 3.74 sentences                      |
| NNJM     | 4.59 hours                      | 3.69 sentences                      |
| BCCNN    | 5.23 hours                      | 3.41 sentences                      |
| SNNJM    | 6.16 hours                      | 3.26 sentences                      |

context. In other words, encoding bilingual context that includes not only target-side context but also source-side context requires more time than encoding monolingual context.

2) For decoding efficiency, it is observed that the proposed SNNJM decreases slightly over NNLM and BCCNN. This is because the SNNJM jointly learns syntax-based context representations and translation prediction instead of using sequential context representation to predict translation in NNJM. Specially, the SNNJM must learn SLSUs and their vector representations except for predicting translations over the learned vector representations of SLSUs.

6.7 Sample Outputs

In this section, we compare the output of our systems and the Baseline. The example in Fig 7 shows the powerful syntactic constraint mechanism of our model for translation prediction.

First, the proposed model translates the source word “lianxu” into “have continuously” instead of “continuously” in the NNJM. The difference is that the proposed model can encode additional source-side phrase structural information compared with the NNJM, which further makes the source word “kaizhan” translated into the target word “conducted” instead of the target word “development” in the NNJM. Second, both of the baseline and the NNJM translate the source word “xingdong” into the target word “operations” which is different from the “Ref”. Instead, our model gives a translation “actions” that is consistent with the “Ref”. These mean that source phrase structural information is beneficial to the translation prediction for SMT. Finally, compared with the baseline and the NNJM, the proposed model can learn phrase structure of “lianxu kaizhang” (have continuously conducted) and “baochi le (kept up), which are important information for correctly predicting target translation “and kept up” of source phrase “baochi le”.

7. Discussion

The seminal work of modeling context with source language information can be traced to bilingual language models [20],[41]. The context of a word is often defined as the words appearing in a fixed-size window context, including target- and source-side context, such as bilingual word pairs [11], minimum translation units [37], bilingual word-based joint translation and reordering models [42]. However, nearly all of these works used traditional n-gram methods to model context, and was therefore subject to the chal-
lenge of data sparsity caused by a mass of bilingual word or phrase pairs [42]. This caused difficulties in the estimation of higher-order n-gram models and lacked NN’s ability to generalize semantically [1].

Recently, many efforts have been initiated on learning vector representation of context with source language information using NN-based methods [3]–[6], [39], [40]. Le et al. [40] proposed a continuous space translation model with neural network to predict each target word given the previous bilingual phrases. Hu et al. [39] modeled minimum translation units using RNNs to predict translation. Devlin et al. [4] extended the target context with “relevant” source word sequences, and used an FFNN to predict translation over their context. Their work was further advanced in [5], [6], which encoded the entire source sentence. Chen et al. [38], [43] flatten local dependency structures including parent, children, and sibling words into a word tuple, which is represented as a compositional vector by a convolutional NN to model the long-distance dependency constraints. In spite of their success, their approaches center around capturing relations between each word and its local dependency words. In spite of their success, these approaches centered on encoding sequential context words for achieving state-of-the-art performance in SMT. They do not take into account explicit syntax information, especially syntax or phrase labels which have been shown to be beneficial for discrete symbol context representations.

Syntactic information, including source syntax trees, has been used in phrase-based SMT. Bach et al. [44] models transition among syntax subtrees during translation, without taking into account syntax and POS tags. Another study exploits source and target syntax trees for phrase-based MT output reranking [45] or defining pre-ordering rules [46], [47], rather than for translation decoding. Garmash et al. [12] expand target word with POS-tagging of source dependency words.

Recently, source syntax information is successfully used to improve the performance of NMT [48]. Typically, these methods learn richer context information for NMT by explicitly encoding the equivalent syntax information of the source input, such as the source phrase structure [49], the POS-tagger and dependency constraints [50]–[52], and source syntax tree [53], [54].

The model proposed in this paper can encode source syntactic information to learn context representation dynamically at different time steps during translation decoding. Especially, a unique TCNN is proposed to learn vector representations for the SLS, thereby greatly alleviating the data sparsity caused by large SLSUs. Actually, the TCNN, which is a structural neural network, can learn syntax-based context more effectively for improving translation prediction. In other words, other similar structural neural networks, for example tree-structured LSTM, are also applicable. We keep the phrase-based translation model as the principal SMT framework and design a new FFNN with a TCNN to predict translation using the learned syntactic context representation.

8. Conclusions and Future Work

In this paper, we explored source syntactic structures across languages to learn more effective context information for translation prediction. To learn context representation, a unique TCNN was proposed to encode context with syntactic information. Experimental results demonstrated that the proposed syntax-based NN joint model can greatly improve translation prediction using syntax-based context sequences.

In the future, we intend to investigate more linguistic knowledge to enhance translation prediction further. Additionally, we aim to integrate our method into NMT.

Acknowledgments

We are grateful to the anonymous reviewers for their insightful comments and suggestions. This work is supported by the National Natural Science Foundation of China via grant 91520204 and the National Key Technologies R&D Program of China via grant 2017YFB1002102.

References

[1] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” Advances in Neural Information Processing Systems 26, ed. C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, pp.3111–3119, Curran Associates, Inc., Dec. 2013.
[2] N. Kalchbrenner and P. Blunsom, “Recurrent continuous translation models,” Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA, pp.1700–1709, Association for Computational Linguistics, Oct. 2013.
[3] M. Auli, M. Galley, C. Quirk, and G. Zweig, “Joint language and translation modeling with recurrent neural networks,” Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA, pp.1044–1054, Association for Computational Linguistics, Oct. 2013.
[4] J. Devlin, R. Zbib, Z. Huang, T. Lamar, R. Schwartz, and J. Makhoul, “Fast and robust neural network joint models for statistical machine translation,” Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, Maryland, pp.1370–1380, Association for Computational Linguistics, June 2014.
[5] J. Zhang, D. Zhang, and J. Hao, “Local translation prediction with global sentence representation,” Proceedings of the 24th International Conference on Artificial Intelligence, pp.1398–1404, AAAI Press, July 2015.
[6] F. Meng, Z. Lu, M. Wang, H. Li, W. Jiang, and Q. Liu, “Encoding source language with convolutional neural network for machine translation,” Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, pp.20–30, Association for Computational Linguistics, July 2015.
[7] D. Xiong, M. Zhang, A. Aw, and H. Li, “Linguistically annotated BTG for statistical machine translation,” Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), Manchester, UK, pp.1009–1016, Coling 2008 Organizing Committee, Aug. 2008.
[8] D. Xiong, M. Zhang, A. Aw, and H. Li, “A syntax-driven bracketing model for phrase-based translation,” Proceedings of the Joint
Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, Suntec, Singapore, pp.315–323, Association for Computational Linguistics, Aug. 2009.

[9] N. Ge, “A direct syntax-driven reordering model for phrase-based machine translation,” Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Los Angeles, California, pp.849–857, Association for Computational Linguistics, June 2010.

[10] R. Yeniterzi and K. Oflazer, “Syntacto-morphology mapping in factored phrase-based statistical machine translation from english to turkish,” Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, pp.454–464, Association for Computational Linguistics, July 2010.

[11] J. Niehues, T. Herrmann, S. Vogel, and A. Waibel, “Wider context by using bilingual language models in machine translation,” Proceedings of the Sixth Workshop on Statistical Machine Translation, Edinburgh, Scotland, pp.198–206, Association for Computational Linguistics, July 2011.

[12] E. Garmash and C. Monz, “Dependency-based bilingual language models for reordering in statistical machine translation,” Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, pp.1689–1700, Association for Computational Linguistics, Oct. 2014.

[13] F.J. Och and H. Ney, “Discriminative training and maximum entropy models for statistical machine translation,” Proceedings of 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pp.295–302, Association for Computational Linguistics, July 2002.

[14] P.F. Brown, V.J.D. Pietra, S.A.D. Pietra, and R.L. Mercer, “The mathematics of statistical machine translation: Parameter estimation,” Computational Linguistics, vol.19, no.2, pp.263–311, June 1993.

[15] Y. Liu, Q. Liu, and S. Lin, “Tree-to-string alignment template for statistical machine translation,” Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Sydney, Australia, pp.609–616, Association for Computational Linguistics, July 2006.

[16] L. Shen, J. Xu, and R. Weischedel, “A new string-to-dependency machine translation algorithm with a target dependency language model,” Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, Columbus, Ohio, pp.577–585, Association for Computational Linguistics, June 2008.

[17] D. Chiang, “Learning to translate with source and target syntax,” Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, pp.1443–1452, Association for Computational Linguistics, July 2010.

[18] P. Koehn, F.J. Och, and D. Marcu, “Statistical phrase-based translation,” Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Stroudsburg, PA, USA, pp.48–54, Association for Computational Linguistics, July 2003.

[19] N. Durran, H. Schmid, and A. Fraser, “A joint sequence translation model with integrated reordering,” Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, pp.1045–1054, Association for Computational Linguistics, June 2011.

[20] J.B. Maríño, R.E. Bancks, 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, Dec. 2006.

[21] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” Journal of Machine Learning Research, vol.12, pp.2493–2537, Nov. 2011.

[22] L. Mou, H. Peng, G. Li, Y. Xu, L. Zhang, and Z. Jin, “Discriminative neural sentence modeling by tree-based convolution,” Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, pp.2315–2325, Association for Computational Linguistics, Sept. 2015.

[23] Y. Kim, “Convolutional neural networks for sentence classification,” Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, pp.1746–1751, Association for Computational Linguistics, Oct. 2014.

[24] A. Vaswani, Y. Zhao, V. Fossum, and D. Chiang, “Decoding with large-scale neural language models improves translation,” Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA, pp.1387–1392, Association for Computational Linguistics, Oct. 2013.

[25] J. Zhang, M. Utiyama, E. Sumita, G. Neubig, and S. Nakamura, “A binarized neural network joint model for machine translation,” Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, pp.2094–2099, Association for Computational Linguistics, Sept. 2015.

[26] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, “A neural probabilistic language model,” Journal of Machine Learning Research, vol.3, pp.1137–1153, March 2003.

[27] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, “Moses: Open source toolkit for statistical machine translation,” Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions, Prague, Czech Republic, pp.177–180, Association for Computational Linguistics, June 2007.

[28] P.-C. Chang, H. Tseng, D. Jurafsky, and C.D. Manning, “Discriminative reordering with Chinese grammatical relations features,” Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation, Boulder, Colorado, pp.51–59, Association for Computational Linguistics, June 2009.

[29] F.J. Och and H. Ney, “A systematic comparison of various statistical alignment models,” Computational Linguistics, vol.29, no.1, pp.19–51, March 2003.

[30] F.J. Och, “Minimum error rate training in statistical machine translation,” Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan, pp.160–167, Association for Computational Linguistics, July 2003.

[31] K. Papineni, S. Roukos, T. Ward, and W.J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” Proceedings of 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pp.311–318, Association for Computational Linguistics, July 2002.

[32] P. Koehn, “Statistical significance tests for machine translation evaluation,” Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, ed. D. Lin and D. Wu, Barcelona, Spain, pp.388–395, Association for Computational Linguistics, July 2004.

[33] R. Sennrich, O. Firat, K. Cho, A. Birch, B. Haddow, J. Hitschler, M. Junczys-Dowmunt, S. Läubli, A.V. Miceli Barone, J. Mokry, and M. Nadejde, “Nematus: a toolkit for neural machine translation,” Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain, pp.65–68, Association for Computational Linguistics, April 2017.

[34] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” CoRR, vol.abs/1207.0580, 2012.

[35] M.D. Zeiler, “ADADELTA: an adaptive learning rate method,” CoRR, vol.abs/1212.5701, 2012.

[36] M. Galley and C.D. Manning, “A simple and effective hierarchical phrase reordering model,” Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Honolulu, Hawaii, pp.848–856, Association for Computational Linguistics, Oct. 2008.
[37] N. Durrani, A. Fraser, H. Schmid, H. Hoang, and P. Koehn, “Can markov models over minimal translation units help phrase-based smt?,” Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, pp.399–405, Association for Computational Linguistics, Aug. 2013.

[38] K. Chen, T. Zhao, M. Yang, L. Liu, A. Tamura, R. Wang, M. Utiyama, and E. Sumita, “A neural approach to source dependency based context model for statistical machine translation,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol.26, no.2, pp.266–280, Feb. 2018.

[39] Y. Hu, M. Auli, Q. Gao, and J. Gao, “Minimum translation modeling with recurrent neural networks,” Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, Gothenburg, Sweden, pp.20–29, Association for Computational Linguistics, April 2014.

[40] H.S. Le, A. Allauzen, and F. Yvon, “Continuous space translation models with neural networks,” Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Montréal, Canada, pp.39–48, Association for Computational Linguistics, June 2012.

[41] J.M. Crego and F. Yvon, “Improving reordering with linguistically informed bilingual n-grams,” Proceedings of the 26th International Conference on Computational Linguistics, Beijing, China, pp.197–205, Coling 2010 Organizing Committee, Aug.t 2010.

[42] A. Guta, T. Alkhouli, J.-T. Peter, J. Wuebker, and H. Ney, “A comparison between count and neural network models based on joint translation and reordering sequences,” Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, pp.1401–1411, Association for Computational Linguistics, September 2015.

[43] K. Chen, T. Zhao, M. Yang, and L. Liu, “Translation prediction with source dependency-based context representation,” Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California USA, July 2017.

[44] N. Bach, Q. Gao, and S. Vogel, “Source-side dependency tree reordering models with subtree movements and constraints,” Ottawa, Canada, International Association for Machine Translation, Aug. 2009.

[45] K. Gimpel and N.A. Smith, “Phrase dependency machine translation with quasi-synchronous tree-to-tree features,” Computational Linguistics, vol.40, no.2, pp.349–401, June 2014.

[46] C. Wang, M. Collins, and P. Koehn, “Chinese syntactic reordering for statistical machine translation,” Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, Czech Republic, pp.737–745, Association for Computational Linguistics, June 2007.

[47] M. Khalilov and J.A.R. Fonollosa, “Syntax-based reordering for statistical machine translation,” Computer Speech and Language, vol.25, no.4, pp.761–788, Oct. 2011.

[48] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” CoRR, vol. abs/1409.0473, 2014.

[49] A. Eriguchi, K. Hashimoto, and Y. Tsuruoka, “Tree-to-sequence attentional neural machine translation,” Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, pp.823–833, Association for Computational Linguistics, Aug. 2016.

[50] R. Sennrich and B. Haddow, “Linguistic input features improve neural machine translation,” Proceedings of the First Conference on Machine Translation, Berlin, Germany, pp.83–91, Association for Computational Linguistics, Aug. 2016.

[51] K. Chen, R. Wang, M. Utiyama, L. Liu, A. Tamura, E. Sumita, and T. Zhao, “Neural machine translation with source dependency representation,” Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, pp.2846–2852, Association for Computational Linguistics, Sept. 2017.

Kehai Chen received the B.S. degree in computer science from Xi’an University of Technology, China in 2010 and the M.S. degree in computer science from Chinese Academy of Sciences, China in 2013. He is currently a Ph.D. candidate in Harbin Institute of Technology. His research interests include machine translation and natural language processing.

Tiejun Zhao is a professor of School of Computer Science and Technology, Harbin Institute of Technology. His research interests include: natural language understanding, content-based web information processing, and applied artificial intelligence. He has published 3 academic books and 60 papers on journals and conferences in recent 3 years. Prof. Zhao has been a PC member on ACL, COLING in current 5 years and was also assigned an MT Track Co-chair on COLING 2014.

Muyun Yang is an associate professor of School of Computer Science and Technology, Harbin Institute of Technology. His research interests include: machine translation, computational linguistics and artificial intelligence. He has published 1 academic book and 20 papers on journals and conferences in recent 3 years.