Sequence-to-Dependency Neural Machine Translation

Shuangzhi Wu[†*], **Dongdong Zhang**[‡], **Nan Yang**[‡], **Mu Li**[‡], **Ming Zhou**[‡]
[†]Harbin Institute of Technology, Harbin, China
[‡]Microsoft Research
{v-shuawu, dozhang, nanya, muli, mingzhou}@microsoft.com

Abstract

Nowadays a typical Neural Machine Translation (NMT) model generates translations from left to right as a linear sequence, during which latent syntactic structures of the target sentences are not explicitly concerned. Inspired by the success of using syntactic knowledge of target language for improving statistical machine translation, in this paper we propose a novel Sequence-to-Dependency Neural Machine Translation (SD-NMT) method, in which the target word sequence and its corresponding dependency structure are jointly constructed and modeled, and this structure is used as context to facilitate word generations. Experimental results show that the proposed method significantly outperforms state-of-the-art baselines on Chinese-English and Japanese-English translation tasks.

1 Introduction

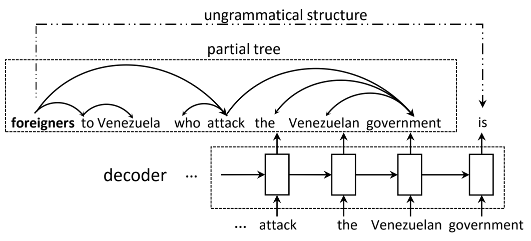
Recently, Neural Machine Translation (NMT) with the attention-based encoder-decoder framework (Bahdanau et al., 2015) has achieved significant improvements in translation quality of many language pairs (Bahdanau et al., 2015; Luong et al., 2015a; Tu et al., 2016; Wu et al., 2016). In a conventional NMT model, an encoder reads in source sentences of various lengths, and transforms them into sequences of intermediate hidden vector representations. After weighted by attention operations, combined hidden vectors are used by the decoder to generate translations. In most of cases, both encoder and decoder are implemented as recurrent neural networks (RNNs).

[*]Contribution during internship at Microsoft Research.

Many methods have been proposed to further improve the sequence-to-sequence NMT model since it was first proposed by Sutskever et al. (2014) and Bahdanau et al. (2015). Previous work ranges from addressing the problem of out-of-vocabulary words (Jean et al., 2015), designing attention mechanism (Luong et al., 2015a), to more efficient parameter learning (Shen et al., 2016), using source-side syntactic trees for better encoding (Eriguchi et al., 2016) and so on. All these NMT models employ a sequential recurrent neural network for target generations. Although in theory RNN is able to remember sufficiently long history, we still observe substantial incorrect translations which violate long-distance syntactic constraints. This suggests that it is still very challenging for a linear RNN to learn models that effectively capture many subtle long-range word dependencies. For example, Figure 1 shows an incorrect translation related to the long-distance dependency. The translation fragment in italic is locally fluent around the word *is*, but from a global view the translation is ungrammatical. Actually, this part of translation should be mostly affected by the distant plural noun *foreigners* rather than words *Venezuelan government* nearby.

Fortunately, such long-distance word correspondence can be well addressed and modeled by syntactic dependency trees. In Figure 1, the head word *foreigners* in the partial dependency tree (top dashed box) can provide correct structural context for the next target word, with this information it is more likely to generate the correct word *will* rather than *is*. This structure has been successfully applied to significantly improve the performance of statistical machine translation (Shen et al., 2008). On the NMT side, introducing target syntactic structures could help solve the problem of ungrammatical output because it can bring two advantages over state-of-the-art NMT models:

a) syntactic trees can be used to model the grammatical validity of translation candidates; b) partial syntactic structures can be used as additional context to facilitate future target word prediction.


ungrammatical structure
partial tree
foreigners to Venezuela who attack the Venezuelan government
is
decoder ...
... attack the Venezuelan government

Source : 他 还 说 , 来 委 外国人 若 攻击 委内瑞拉 政府 会 面临 严重 后果 , 将 被 驱逐出境 .

Ref : He added that foreign visitors to Venezuela who criticize the Venezuelan government will face serious consequences and will be deported .

NMT : He also said that foreigners to Venezuela who attack the Venezuelan government *is facing serious consequences*, will be deported .

Figure 1: Dependency trees help the prediction of the next target word. “NMT” refers to the translation result from a conventional NMT model, which fails to capture the long distance word relation denoted by the dashed arrow.

However, it is not trivial to build and leverage syntactic structures on the target side in current NMT framework. Several practical challenges arise:

(1) How to model syntactic structures such as dependency parse trees with recurrent neural network;

(2) How to efficiently perform both target word generation and syntactic structure construction tasks simultaneously in a single neural network;

(3) How to effectively leverage target syntactic context to help target word generation.

To address these issues, we propose and empirically evaluate a novel Sequence-to-Dependency Neural Machine Translation (SD-NMT) model in our paper. An SD-NMT model encodes source inputs with bi-directional RNNs and associates them with target word prediction via attention mechanism as in most NMT models, but it comes with a new decoder which is able to jointly generate target translations and construct their syntactic dependency trees. The key difference from conventional NMT decoders is that we use two RNNs, one for translation generation and the other for dependency parse tree construction, in which incremental parsing is performed with the arc-standard shift-reduce algorithm proposed by Nivre (2004). We will describe in detail how these two RNNs work interactively in Section 3.

We evaluate our method on publicly available data sets with Chinese-English and Japanese-English translation tasks. Experimental results show that our model significantly improves translation accuracy over the conventional NMT and SMT baseline systems.

2 Background

2.1 Neural Machine Translation

As a new paradigm to machine translation, NMT is an end-to-end framework (Sutskever et al., 2014; Bahdanau et al., 2015) which directly models the conditional probability $P(Y|X)$ of target translation $Y = y_1, y_2, ..., y_n$ given source sentence $X = x_1, x_2, ..., x_m$. An NMT model consists of two parts: an encoder and a decoder. Both of them utilize recurrent neural networks which can be a Gated Recurrent Unit (GRU) (Cho et al., 2014) or a Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) in practice. The encoder bidirectionally encodes a source sentence into a sequence of hidden vectors $H = h_1, h_2, ..., h_m$ with a forward RNN and a backward RNN. Then the decoder predicts target words one by one with probability

$$P(Y|X) = \prod_{j=1}^{n} P(y_j|y_{<j}, H) \tag{1}$$

Typically, for the jth target word, the probability $P(y_j|y_{<j}, H)$ is computed as

$$P(y_j|y_{<j}, H) = g(s_j, y_{j-1}, c_j) \tag{2}$$

where g is a nonlinear function that outputs the probability of y_j, and s_j is the RNN hidden state. The context c_j is calculated at each timestamp j based on H by the attention network

$$\begin{aligned} c_j &= \sum_{k=1}^{m} a_{jk} h_k & (3) \\ a_{jk} &= \frac{\exp(e_{jk})}{\sum_{i=1}^{m} \exp(e_{ji})} & (4) \\ e_{jk} &= v_a^T \tanh(W_a s_{j-1} + U_a h_k) & (5) \end{aligned}$$

where v_a, W_a, U_a are the weight matrices. The attention mechanism is effective to model the correspondences between source and target.

2.2 Dependency Tree Construction

We use a shift-reduce transition-based dependency parser to build the syntactic structure for the target language in our work. Specially, we adopt the arc-standard algorithm (Nivre, 2004) to perform incremental parsing during the translation process. In this algorithm, a stack and a buffer are maintained to store the parsing state over which three kinds of transition actions are applied. Let w_0 and w_1 be two topmost words in the stack, and $\bar{w}$ be the current new word in a sequence of input, three transition actions are described as below.

- Shift(SH) : Push $\bar{w}$ to the stack.
- Left-Reduce(LR(d)) : Link w_0 and w_1 with dependency label d as $w_0 \overset{d}{\rightarrow} w_1$, and reduce them to the head w_0.
- Right-Reduce(RR(d)) : Link w_0 and w_1 with dependency label d as $w_0 \overset{d}{\leftarrow} w_1$, and reduce them to the head w_1.

During parsing, an specific structure is used to record the dependency relationship between different words of input sentence. The parsing finishes when the stack is empty and all input words are consumed. As each word must be pushed to the stack once and popped off once, the number of actions needed to parse a sentence is always $2n$, where n is the length of the sentence (Nivre, 2004). Because each valid transition action sequence corresponds to a unique dependency tree, a dependency tree can also be equivalently represented by a sequence of transition actions.

3 Sequence-to-Dependency Neural Machine Translation

An SD-NMT model is an extension to the conventional NMT model augmented with syntactic structural information of target translation. Given a source sentence $X = x_1,x_2,..,x_m$, its target translation $Y = y_1,y_2,..,y_n$ and Y's dependency parse tree T, the goal of the extension is to enable us to compute the joint probability $P(Y,T|X)$. As in most structural learning tasks, the full prediction of Y and T is further decomposed into a chain of smaller predictions. For translation Y, it is generated in the left-to-right order as $y_1, y_2, .., y_n$ following the way in a normal sequence-to-sequence model. For Y's parse tree T, instead of directly modeling the tree itself, we predict a parsing action sequence A which can map Y to T. Thus at top level our SD-NMT model can be formulated as

$$\begin{aligned} P(Y,T|X) &= P(Y,A|X) \\ &= P(y_1y_2..y_n, a_1, a_2..a_l|X) \end{aligned} \quad (6)$$

where $A = a_1,a_2,..,a_j,..,a_l$ [1] with length l ($l = 2n$), $a_j \in \{\text{SH}, \text{RR}(d), \text{LR}(d)\}$[2].

Two recurrent neural networks, Word-RNN and Action-RNN, are used to model generation processes of translation sequence Y and parsing action sequence A respectively. Figure 2 shows an example how translation Y and its parsing actions are predicted step by step.

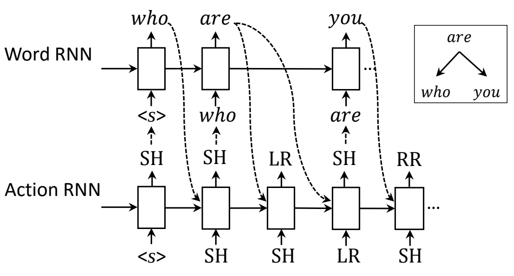


Figure 2: Decoding example of our SD-NMT model for target sentence "who are you" with transition action sequence "SH SH LR SH RR". The ending symbol EOS is omitted.

Because the lengths of Word-RNN and Action-RNN are different, they are designed to work in a mutually dependent way: a target word is only allowed to be generated when the SH action is predicted in the action sequence. In this way, we can perform incremental dependency parsing for translation Y and at the same time track the partial parsing status through the translation generation process.

For notational clarity, we introduce a virtual translation sequence $\hat{Y} = \hat{y}_1,\hat{y}_2,..,\hat{y}_j,..,\hat{y}_l$ for Word-RNN which has the same length l with transition action sequence. $\hat{y}_j$ is defined as

$$\hat{y}_j = \begin{cases} y_{v_j} & \delta(\text{SH}, a_j) = 1 \\ y_{v_{j-1}} & \delta(\text{SH}, a_j) = 0 \end{cases}$$

where $\delta(\text{SH}, a_j)$ is 1 when $a_j = \text{SH}$, otherwise 0. v_j is the index of Y, computed by $v_j = \sum_{i=1}^{j} \delta(SH, a_i)$. Apparently the mapping from $\hat{Y}$

[1]In the rest of this paper, a_j represents the transition action, rather than the attention weight in Equation 4.

[2]RR(d) refers to a set of RR actions augmented with dependency labels so as to LR(d).

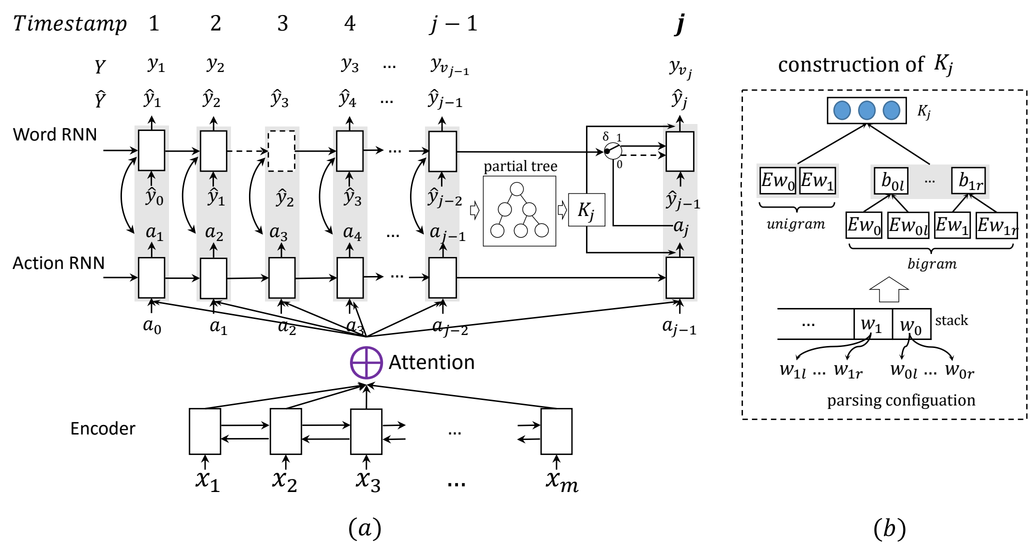


Figure 3: (a) is the overview of SD-NMT model. The dashed arrows mean copying previous recurrent state or word. The two RNNs use the same source context for prediction. $a_j \in \{\text{SH}, \text{RR}(d), \text{LR}(d)\}$. The bidirection arrow refers to the interaction between two RNNs. (b) shows the construction of syntactic context. The gray box means the concatenation of vectors

to Y is deterministic, and Y can be easily derived given $\hat{Y}$ and A.

With the notation of $\hat{Y}$, the sequence probability of Y and A can be written as

$$P(A|X,\hat{Y}_{<l}) = \prod_{j=1}^{l} P(a_j|a_{<j},X,\hat{Y}_{<j}) \tag{7}$$

$$P(\hat{Y}|X,A_{\le l}) = \prod_{j=1}^{l} P(\hat{y}_j|\hat{y}_{<j},X,A_{\le j})^{\delta(SH,a_j)} \tag{8}$$

where $\hat{Y}_{<j}$ refers to the subsequence $\hat{y}_1,\hat{y}_2,..,\hat{y}_{j-1}$, and $A_{\le j}$ to $a_1,a_2,..,a_j$. Based on Equation 7 and 8, the overall joint model can be computed as

$$P(Y,T|X) = P(A|X,\hat{Y}_{<l}) \times P(\hat{Y}|X,A_{\le l}) \tag{9}$$

As we have two RNNs in our model, the termination condition is also different from a conventional NMT model. In decoding, we maintain a stack to track the parsing configuration, and our model terminates once the Word-RNN predicts a special ending symbol `EOS` and all the words in the stack have been reduced.

Figure 3 (a) gives an overview of our SD-NMT model. Due to space limitation, the detailed interconnections between two RNNs are only illustrated at timestamp j. The encoder of our model follows standard bidirectional RNN configuration. At timestamp j during decoding, our model first predicts an action a_j by Action-RNN, then Word-RNN checks the condition gate δ according to a_j. If $a_j = \text{SH}$, the Word-RNN will generate a new state (solid arrow) and predict a new target word y_{v_j}, otherwise it just copies previous state (dashed arrow) to the current state. For example, at timestamp 3, $a_3 \neq \text{SH}$, the state of Word-RNN is copied from its previous one. Meanwhile, $\hat{y}_3 = y_2$ is used as the immediate proceeding word in translation history.

When computing attention scores, we extend Equation 5 by replacing the decoder hidden state with the concatenation of Word-RNN hidden state s and Action-RNN hidden state s' (gray boxes in Figure 3). The new attention score is then updated as

$$e_{jk} = v_a^T \tanh(W_a[s_{j-1}; s'_{j-1}] + U_a h_k) \tag{10}$$

3.1 Syntactic Context for Target Word Prediction

Syntax has been proven useful for sentence generation task (Dyer et al., 2016). We propose to leverage target syntax to help translation generation. In our model, the syntactic context K_j at timestamp j is defined as a vector which is computed by a feed-forward network based on current parsing configuration of Action-RNN. Denote that w_0 and w_1 are two topmost words in the stack, w_{0l} and w_{1l} are their leftmost modifiers in the partial tree, w_{0r} and w_{1r} their rightmost modifiers respectively. We define two unigram features and four bigram features. The unigram features are w_0 and w_1 which are represented by the word embedding vectors. The bigram features are $w_0 w_{0l}$, $w_0 w_{0r}$, $w_1 w_{1l}$ and $w_1 w_{1r}$. Each of them is computed by $b_{hc} = \tanh(W_b E w_h + U_b E w_{hc})$, $h \in \{0, 1\}$, $c \in \{l, r\}$. These kinds of feature template have beeb proven effective in dependency parsing task (Zhang and Clark, 2008). Based on these features, the syntactic context vector K_j is computed as

$$K_j = \tanh(W_k[Ew_0; Ew_1] + U_k[b_{0l}; b_{0r}; b_{1l}; b_{1r}]) \tag{11}$$

where W_k, U_k, W_b, U_b are the weight matrices, E stands for the embedding matrix. Figure 2 (b) gives an overview of the construction of K_j. Note that zero vector is used for padding the words which are not available in the partial tree, so that all the K vectors have the same input size in computation.

Adding K_j to Equation 2, the probabilities of transition action and word in Equation 7 and 8 are then updated as

$$P(a_j | a_{<j}, X, \hat{Y}_{<j}) = g(s'_j, a_{j-1}, c_j, K_j) \tag{12}$$

$$P(\hat{y}_j | \hat{y}_{<j}, X, A_{\leq j}) = g(s_j, \hat{y}_{j-1}, c_j, K_j) \tag{13}$$

After each prediction step in Word-RNN and Action-RNN, the syntax context vector K will be updated accordingly. Note that K is not used to calculate the recurrent states s in this work.

3.2 Model Training and Decoding

For SD-NMT model, we use the sum of log-likelihoods of word sequence and action sequence as objective function for training algorithm, so that the joint probability of target translations and their parsing trees can be maximized:

$$J(\theta) = \sum_{(X,Y,A) \in D} \log P(A | X, \hat{Y}_{<l}) + \log P(\hat{Y} | X, A_{\leq l}) \tag{14}$$

We also use mini-batch for model training. As the target dependency trees are known in the bilingual corpus during training, we pre-compute the partial tree state and syntactic context at each time stamp for each training instance. Thus it is easy for the model to process multiple trees in one batch.

In the decoding process of an SD-NMT model, the score of each search path is the sum of log probabilities of target word sequence and transition action sequence normalized by the sequence length:

$$\text{score} = \frac{1}{l} \sum_{j=1}^{l} \log P(a_j | a_{<j}, X, \hat{Y}_{<j}) + \frac{1}{n} \sum_{j=1}^{l} \delta(SH, a_j) \log P(\hat{y}_j | \hat{y}_{<j}, X, A_{\leq j}) \tag{15}$$

where n is word sequence length and l is action sequence length.

4 Experiments

The experiments are conducted on the Chinese-English task as well as the Japanese-English translation tasks where the same data set from WAT 2016 ASPEC corpus (Nakazawa et al., 2016) [3] is used for a fair comparison with other work. In addition to evaluate translation performance, we also investigate the quality of dependency parsing as a by-product and the effect of parsing quality against translation quality.

4.1 Setup

In the Chinese-English task, the bilingual training data consists of a set of LDC datasets, [4] which has around 2M sentence pairs. We use NIST2003 as the development set, and the testsets contain NIST2005, NIST2006, NIST2008 and NIST2012. All English words are lowercased.

In the Japanese-English task, we use top 1M sentence pairs from ASPEC Japanese-English corpus. The development data contains 1,790 sentences, and the test data contains 1,812 sentences with single reference per source sentence.

To train SD-NMT model, the target dependency tree references are needed. As there is no golden annotation of parse trees over the target training data, we use pseudo parsing results as the target dependency references, which are got from an in-house developed arc-eager dependency parser based on work in (Zhang and Nivre, 2011).

[3] http://orchid.kuee.kyoto-u.ac.jp/ASPEC/

[4] LDC2003E14, LDC2005T10, LDC2005E83, LDC2006E26, LDC2006E34, LDC2006E85, LDC2006E92, LDC2003E07, LDC2002E18, LDC2005T06, LDC2003E07, LDC2004T07, LDC2004T08, LDC2005T06

Settings	NIST 2005	NIST 2006	NIST 2008	NIST 2012	Average
HPSMT	35.34	33.56	26.06	27.47	30.61
RNNsearch	38.07	38.95	31.61	28.95	34.39
SD-NMT\K	38.83	39.23	31.92	29.72	34.93
SD-NMT	**39.38**	**41.81**	**33.06**	**31.43**	36.42

Table 1: Evaluation results on Chinese-English translation task with BLEU% metric. The "Average" column is the averaged result of all test sets. The numbers in bold indicate statistically significant difference ($p < 0.05$) from baselines.

In the neural network training, the vocabulary size is limited to 30K high frequent words for both source and target languages. All low frequent words are normalized into a special token `unk` and post-processed by following the work in (Luong et al., 2015b). The size of word embedding and transition action embedding is set to 512. The dimensions of the hidden states for all RNNs are set to 1024. All model parameters are initialized randomly with Gaussian distribution (Glorot and Bengio, 2010) and trained on a NVIDIA Tesla K40 GPU. The stochastic gradient descent (SGD) algorithm is used to tune parameters with a learning rate of 1.0. The batch size is set to 96. In the update procedure, Adadelta (Zeiler, 2012) algorithm is used to automatically adapt the learning rate. The beam sizes for both word prediction and transition action prediction are set to 12 in decoding.

The baselines in our experiments are a phrasal system and a neural translation system, denoted by HPSMT and RNNsearch respectively. HPSMT is an in-house implementation of the hierarchical phrase-based model (Chiang, 2005), where a 4-gram language model is trained using the modified Kneser-Ney smoothing (Kneser and Ney, 1995) algorism over the English Gigaword corpus (LDC2009T13) plus the target data from the bilingual corpus. RNNsearch is an in-house implementation of the attention-based neural machine translation model (Bahdanau et al., 2015) using the same parameter settings as our SD-NMT model including word embedding size, hidden vector dimension, beam size, as well as the same mechanism for OOV word processing.

The evaluation results are reported with the case-insensitive IBM BLEU-4 (Papineni et al., 2002). A statistical significance test is performed using the bootstrap resampling method proposed by Koehn (2004) with a 95% confidence level. For Japanese-English task, we use the official evaluation procedure provided by WAT 2016.[5], where both BLEU and RIBES (Isozaki et al., 2010) are used for evaluation.

4.2 Evaluation on Chinese-English Translation

We evaluate our method on the Chinese-English translation task. The evaluation results over all NIST test sets against baselines are listed in Table 1. Generally, RNNsearch outperforms HPSMT by 3.78 BLEU points on average while SD-NMT surpasses RNNsearch 2.03 BLUE point gains on average, which shows that NMT models usually achieve better results than SMT models, and our proposed sequence-to-dependency NMT model performs much better than traditional sequence-to-sequence NMT model.

We also investigate the effect of syntactic knowledge context by excluding its computation in Equation 12 and 13. The alternative model is denoted by SD-NMT\K. According to Table 1, SD-NMT\K outperforms RNNsearch by 0.54 BLEU points but degrades SD-NMT by 1.49 BLEU points on average, which demonstrates that the long distance dependencies captured by the target syntactic knowledge context, such as leftmost/rightmost children together with their dependency relationships, really bring strong positive effects on the prediction of target words.

In addition to translation quality, we compare the perplexity (PPL) changes on the development set in terms of numbers of training mini-batches for RNNsearch and SD-NMT in Figure 4. We can see that the PPL of SD-NMT is initially higher than that of RNNsearch, but decreased to be lower over time. This is mainly because the quality of parse tree is too poor at the beginning which degrades translation quality and leads to higher PPL. After some training iterations, the SD-NMT

[5]http://lotus.kuee.kyoto-u.ac.jp/WAT/evaluation/index.html

	BLEU	RIBES	System Description
SMT Hiero	18.72	0.6511	Moses' Hierarchical Phrase-based SMT
SMT Phrase	18.45	0.6451	Moses' Phrase-based SMT
SMT S2T	20.36	0.6782	Moses' String-to-Tree Syntax-based SMT
Cromieres (2016)(Single model)	22.86	-	Single-layer NMT model without ensemble
Cromieres (2016)(Self-ensemble)	24.71	0.7508	Self-ensemble of 2-layer NMT model
Cromieres (2016)(4-Ensemble)	26.22	0.7566	Ensemble of 4 single-layer NMT models
RNNsearch	23.50	0.7459	Single-layer NMT model
SD-NMT	25.93	0.7540	Single-layer SD-NMT model

Table 2: Evaluation results on Japanese-English translation task.

model learns reasonable inferences of parse trees which begins to help target word generation and leads to lower PPL.

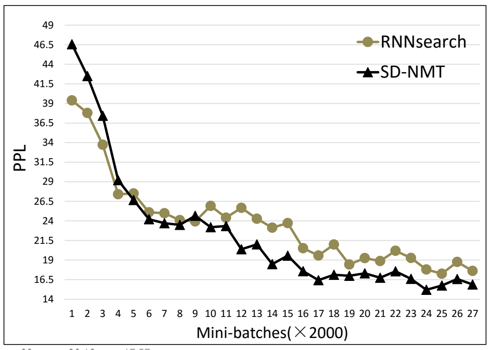


Figure 4: Perplexity (PPL) changes in terms of numbers of training mini-batches.

In our experiments, the time cost of SD-NMT is two times of that for RNNsearch due to a more complicated model structure. But we think it is a worthy trade to pursue high quality translations.

4.3 Evaluation on Japanese-English Translation

In this section, we report results on the Japanese-English translation task. To ensure fair comparisons, we use the same training data and follow the pre-processing steps recommended in WAT 2016[6]. Table 2 shows the comparison results from 8 systems with the evaluation metrics of BLEU and RIBES. The results in the first 3 rows are produced by SMT systems taken from the official WAT 2016. The remaining results are produced by NMT systems, among which the bottom two row results are taken from our in-house NMT systems and others refer to the work in (Cromieres, 2016; Cromieres et al., 2016) that are the competitive NMT results on WAT 2016. According to Table 2, NMT results still outperform SMT results similar to our Chinese-English evaluation results. The SD-NMT model significantly outperforms most other NMT models, which shows that our proposed approach to modeling target dependency tree benefit NMT systems since our RNNsearch baseline achieves comparable performance with the single layer attention-based NMT system in (Cromieres, 2016). Note that our SD-NMT gets comparable results with the 4 single-layer ensemble model in (Cromieres, 2016; Cromieres et al., 2016). We believe SD-NMT can get more improvements with an ensemble of multiple models in future experiments.

4.4 Effect of the Parsing Accuracy upon Translation Quality

The interaction effect between dependency tree conduction and target word generation is investigated in this section. The experiments are conducted on the Chinese-English task over multiple test sets. We evaluate how the quality of dependency trees affect the performance of translation. In the decoding phase of SD-NMT, beam search is applied to the generations of both transition and actions as illustrated in Equation 15. Intuitively, the larger the beam size of action prediction is, the better the dependency tree quality is. We fix the beam size for generating target words to 12, and change the beam size for action prediction to see the difference. Figure 5 shows the evaluation results of all test sets. There is a tendency for BLEU scores to increase with the growth of action prediction beam size. The reason is that the translation quality increases as the quality of dependency tree improves, which shows the construction of dependency trees can boost the generation of target

[6]http://lotus.kuee.kyoto-u.ac.jp/WAT/baseline/dataPreparationJE.html

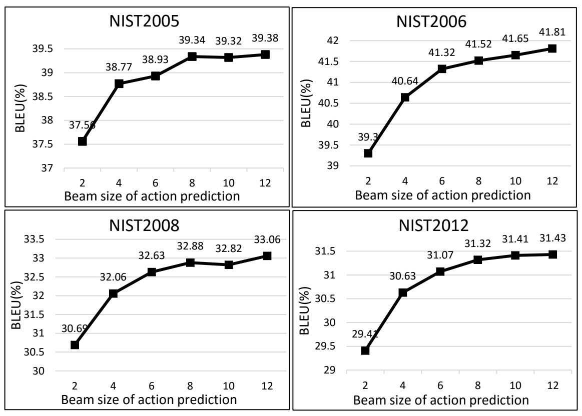


Figure 5: Translation performance against the beam size of action prediction.

words, and vice versa we believe.

4.5 Quality Estimation of Dependency Tree Construction

As a by-product, the quality of dependency trees not only affects the performance of target word generation, but also influences the possible downstream processors or tasks such as text analyses. The direct evaluation of tree quality is not feasible due to the unavailable golden references. So we resort to estimating the consistency between the by-products and the parsing results of our standalone dependency parser with state-of-the-art performance. The higher the consistency is, the closer the performance of by-product is to the standalone parser. To reduce the influence of ill-formed data as much as possible, we build the evaluation data set by heuristically selecting 360 SD-NMT translation results together with their dependency trees from NIST test sets where both source- and target-side do not contain `unk` and have a length of 20-30. We then take the parsing results of the stand-alone parser for these translations as references to indirectly estimate the quality of by-products. We get a UAS (unlabeled attachment score) of 94.96% and a LAS (labeled attachment score) of 93.92%, which demonstrates that the dependency trees produced by SD-NMT are much similar with the parsing results from the standalone parser.

4.6 Translation Example

In this section, we give a case study to explain how our method works. Figure 6 shows a translation example from the NIST testsets. SMT and RNNsearch refer to the translation results from the baselines HPSMT and NMT. For our SD-NMT model, we list both the generated translation and its corresponding dependency tree. We find that the translation of SMT is disfluent and ungrammatical, whereas RNNsearch is better than SMT. Although the translation of RNNsearch is locally fluent around word "have" in the rectangle, both its grammar is incorrect and its meaning is inaccurate from a global view. The word "have" should be in a singular form as its subject is "safety" rather than "workers". For our SD-NMT model, we can see that the translation is much better than baselines and the dependency tree is reasonable. The reason is that after generating the word "workers", the previous subtree in the gray region is transformed to the syntactic context which can guide the generation of the next word as illustrated by the dashed arrow. Thus our model is more likely to generate the correct verb "is" with singular form. In addition, the global structure helps the model correctly identify the inverted sentence pattern of the former translated part and make better choices for the future translation ("only when .. can .." in our translation, "only when .. will .." in the reference), which remains a challenge for conventional NMT model.

5 Related Work

Incorporating linguistic knowledge into machine translation has been extensively studied in Statistic Machine Translation (SMT) (Galley et al., 2006; Shen et al., 2008; Liu et al., 2006). Liu et al. (2006) proposed a tree-to-string alignment template for SMT to leverage source side syntactic information. Shen et al. (2008) proposed a target dependency language model for SMT to employ target-side structured information. These methods show promising improvement for SMT.

Recently, neural machine translation (NMT) has achieved better performance than SMT in many language pairs (Luong et al., 2015a; Zhang et al., 2016; Shen et al., 2016; Wu et al., 2016; Neubig, 2016). In a vanilla NMT model, source and target sentences are treated as sequences where the syntactic knowledge of both sides is neglected. Some effort has been done to incorporate source syntax into NMT. Eriguchi et al. (2016) proposed a tree-to-sequence attentional NMT model where source-side parse tree was used and achieved promising improvement. Intuitively, adding source syntactic information to

[**Source**] 只有 施工 人员 的 安全 得到 了 保证 , 才能 继续 施 工 .

[**Reference**] only when the safety of the workers is guaranteed will they continue with the project .

[**HPSMT**] only safety is assured of construction personnel , to continue construction .

[**RNNsearch**] only when the safety of construction workers *have* been guaranteed *to continue construction* .

[**SD-NMT**] only when the safety of the workers is guaranteed can we continue to work .

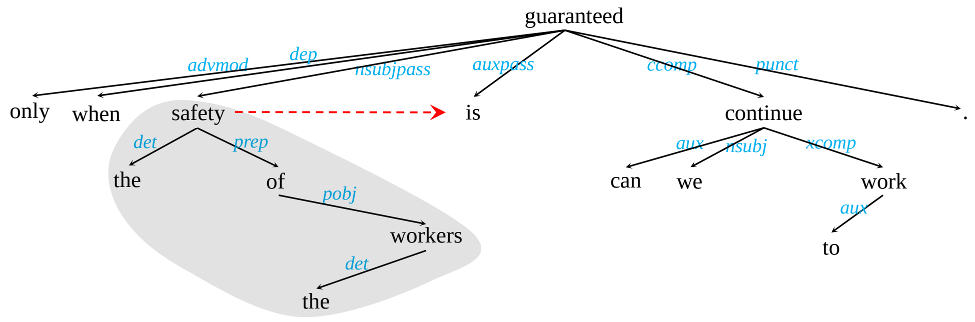


Figure 6: Translation examples of SMT, RNNsearch and our SD-NMT on Chinese-English translation task. The italic words on the arrows are dependency labels. The ending symbol EOS is omitted. RNNsearch fails to capture the long dependency which leads to an ungrammatical result. Whereas with the help of the syntactic tree, our SD-NMT can get a much better translation.

NMT is straightforward, because the source sentence is definitive and easy to attach extra information. However, it is non-trivial to add target syntax as target words are uncertain in decoding process. Up to now, there is few work that attempts to build and leverage target syntactic information for NMT.

There has been work that incorporates syntactic information into NLP tasks with neural networks. Dyer et al. (2016) presented a RNN grammar for parsing and language modeling. They replaced SH with a set of generative actions to generate words under a Stack LSTM framework (Dyer et al., 2015), which achieves promising results for language modeling on the Penn Treebank data. In our work, we propose to involve target syntactic trees into NMT model to jointly learn target translation and dependency parsing where target syntactic context over the parse tree is used to improve the translation quality.

6 Conclusion and Future Work

In this paper, we propose a novel string-to-dependency translation model over NMT. Our model jointly performs target word generation and arc-standard dependency parsing. Experimental results show that our method can boost the two procedures and achieve significant improvements on the translation quality of NMT systems.

In future work, along this research direction, we will try to integrate other prior knowledge, such as semantic information, into NMT systems. In addition, we will apply our method to other sequence-to-sequence tasks, such as text summarization, to verify the effectiveness.

Acknowledgments

We are grateful to the anonymous reviewers for their insightful comments. We also thank Shujie Liu and Zhirui Zhang for the helpful discussions.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR 2015* .

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL 2005*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of ENMLP 2014*.

Fabien Cromieres. 2016. Kyoto-nmt: a neural machine translation implementation in chainer. In *Proceedings of COLING 2016*.

Fabien Cromieres, Chenhui Chu, Toshiaki Nakazawa, and Sadao Kurohashi. 2016. Kyoto university participation to wat 2016. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*. pages 166–174.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of ACL 2015*.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the NAACL 2016*.

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of ACL 2016*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of ACL 2006*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*. volume 9, pages 249–256.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8).

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proceedings of EMNLP*.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of ACL 2015*.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*. IEEE, volume 1, pages 181–184.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *EMNLP*. Citeseer, pages 388–395.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of ACL 2006*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015a. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP 2015*.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015b. Addressing the rare word problem in neural machine translation. In *Proceedings of ACL 2015*.

Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. 2016. Aspec: Asian scientific paper excerpt corpus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association (ELRA), Portoroz, Slovenia, pages 2204–2208.

Graham Neubig. 2016. Lexicons and minimum risk training for neural machine translation: NAIST-CMU at WAT2016. In *Proceedings of the 3nd Workshop on Asian Translation (WAT2016)*. Osaka, Japan.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL 2002*.

Libin Shen, Jinxi Xu, and Ralph M Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *ACL*. pages 577–585.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of ACL 2016*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of ACL 2016*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* .

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .

Biao Zhang, Deyi Xiong, jinsong su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. In *Proceedings of EMNLP 2016*.

Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *EMNLP2008*.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of ACL 2011*.