Modeling Source Syntax for Neural Machine Translation

Junhui Li[†] Deyi Xiong[†] Zhaopeng Tu[‡]
Muhua Zhu[‡] Min Zhang[†] Guodong Zhou[†]
[†]School of Computer Science and Technology,
Soochow University, Suzhou, China
{lijunhui, dyxiong, minzhang, gdzhou}@suda.edu.cn
[‡]Tencent AI Lab, Shenzhen, China
tuzhaopeng@gmail.com, muhuazhu@tencent.com

Abstract

Even though a linguistics-free sequence to sequence model in neural machine translation (NMT) has certain capability of implicitly learning syntactic information of source sentences, this paper shows that source syntax can be explicitly incorporated into NMT effectively to provide further improvements. Specifically, we linearize parse trees of source sentences to obtain structural label sequences. On the basis, we propose three different sorts of encoders to incorporate source syntax into NMT: 1) *Parallel RNN* encoder that learns word and label annotation vectors parallelly; 2) *Hierarchical RNN* encoder that learns word and label annotation vectors in a two-level hierarchy; and 3) *Mixed RNN* encoder that stitchingly learns word and label annotation vectors over sequences where words and labels are mixed. Experimentation on Chinese-to-English translation demonstrates that all the three proposed syntactic encoders are able to improve translation accuracy. It is interesting to note that the simplest RNN encoder, i.e., *Mixed RNN* encoder yields the best performance with an significant improvement of 1.4 BLEU points. Moreover, an in-depth analysis from several perspectives is provided to reveal how source syntax benefits NMT.

1 Introduction

Recently the sequence to sequence model (seq2seq) in neural machine translation (NMT) has achieved certain success over the state-of-the-art of statistical machine translation (SMT) on various language pairs (Bahdanau et al., 2015; Jean et al., 2015; Luong et al., 2015; Luong and Manning, 2015). However, Shi et al. (2016) show that the seq2seq model still fails to capture a lot of deep structural details, even though it is capable of learning certain implicit source syntax from sentence-aligned parallel corpus. Moreover, it requires an additional parsing-task-specific training mechanism to recover the hidden syntax in NMT. As a result, in the absence of explicit linguistic knowledge, the seq2seq model in NMT tends to produce translations that fail to well respect syntax. In this paper, we show that syntax can be well exploited in NMT explicitly by taking advantage of source-side syntax to improve the translation accuracy.

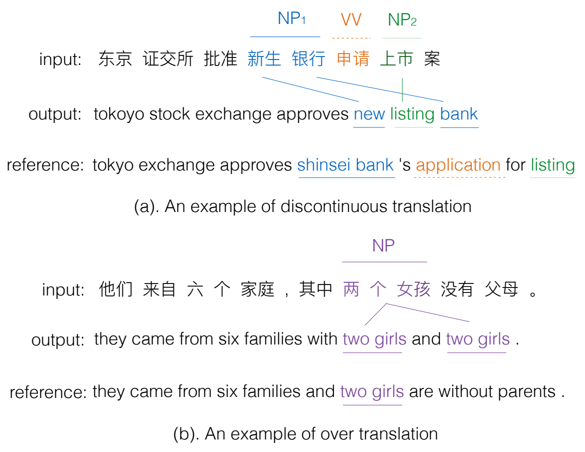
NP1
VV
NP2
input: 东京 证交所 批准 新生 银行 申请 上市 案
output: tokoyo stock exchange approves new listing bank
reference: tokyo exchange approves shinsei bank 's application for listing
(a). An example of discontinuous translation
NP
input: 他们 来自 六 个 家庭 , 其中 两 个 女孩 没有 父母 。
output: they came from six families with two girls and two girls .
reference: they came from six families and two girls are without parents .
(b). An example of over translation

Figure 1: Examples of NMT translation that fail to respect source syntax.

In principle, syntax is a promising avenue for translation modeling. This has been verified by tremendous encouraging studies on syntax-based SMT that substantially improves translation by integrating various kinds of syntactic knowledge (Liu et al., 2006; Marton and Resnik, 2008; Shen et al., 2008; Li et al., 2013). While it is yet to be seen how syntax can benefit NMT effectively, we find that translations of NMT sometimes fail to well respect source syntax. Figure 1 (a) shows a Chinese-to-English translation example of NMT. In this example, the NMT seq2seq model incorrectly translates the Chinese noun phrase (i.e., 新生/*xinsheng* 银行/*yinhang*) into a discontinuous phrase in English (i.e., *new ... bank*) due to the failure of capturing the internal syntactic structure in the input Chinese sentence. Statistics on our development set show that one forth of Chinese noun phrases are translated into discontinuous phrases in English, indicating the substantial disrespect of syntax in NMT translation.[1] Figure 1 (b) shows another example with over translation, where the noun phrase 两/*liang* 个/*ge* 女孩/*nvhai* is translated twice in English. Similar to discontinuous translation, over translation usually happens along with the disrespect of syntax which results in the repeated translation of the same source words in multiple positions of the target sentence.

In this paper we are not aiming at solving any particular issue, either the discontinuous translation or the over translation. Alternatively, we address how to incorporate explicitly the source syntax to improve the NMT translation accuracy with the expectation of alleviating the issues above in general. Specifically, rather than directly assigning each source word with manually designed syntactic labels, as Sennrich and Haddow (2016) do, we linearize a phrase parse tree into a *structural label sequence* and let the model automatically learn useful syntactic information. On the basis, we systematically propose and compare several different approaches to incorporating the label sequence into the seq2seq NMT model. Experimentation on Chinese-to-English translation demonstrates that all proposed approaches are able to improve the translation accuracy.

2 Attention-based NMT

As a background and a baseline, in this section, we briefly describe the NMT model with an attention mechanism by Bahdanau et al. (2015), which mainly consists of an encoder and a decoder, as shown in Figure 2.

Encoder The encoding of a source sentence is formulated using a pair of neural networks, i.e., two

[1]Manually examining 200 random such discontinuously translated noun phrases, we find that 90% of them should be continuously translated according to the reference translation.

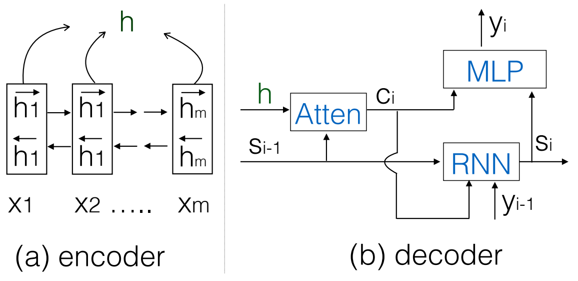

Figure 2: Attention-based NMT model.

recurrent neural networks (denoted *bi-RNN*): one reads an input sequence $x = (x_1, ..., x_m)$ from left to right and outputs a forward sequence of hidden states $(\overrightarrow{h_1}, ..., \overrightarrow{h_m})$, while the other operates from right to left and outputs a backward sequence $(\overleftarrow{h_1}, ..., \overleftarrow{h_m})$. Each source word x_j is represented as h_j (also referred to as word annotation vector): the concatenation of hidden states $\overrightarrow{h_j}$ and $\overleftarrow{h_j}$. Such bi-RNN encodes not only the word itself but also its left and right context, which can provide important evidence for its translation.

Decoder The decoder is also an RNN that predicts a target sequence $y = (y_1, ..., y_n)$. Each target word y_i is predicted via a multi-layer perceptron (MLP) component which is based on a recurrent hidden state s_i, the previous predicted word y_{i-1}, and a source-side context vector c_i. Here, c_i is calculated as a weighted sum over source annotation vectors $(h_1, ..., h_m)$. The weight vector $\alpha_i \in \mathbb{R}^m$ over source annotation vectors is obtained by an attention model, which captures the correspondences between the source and the target languages. The attention weight α_{ij} is computed based on the previous recurrent hidden state s_{i-1} and source annotation vector h_j.

3 NMT with Source Syntax

The conventional NMT models treat a sentence as a sequence of words and ignore external knowledge, failing to effectively capture various kinds of inherent structure of the sentence. To leverage external knowledge, specifically the syntax in the source side, we focus on the parse tree of a sentence and propose three different NMT models that explicitly consider the syntactic structure into encoding. Our purpose is to inform the NMT model the structural context of each word in its corresponding parse tree with the goal that the learned annotation vectors $(h_1, ..., h_m)$ encode not only the information of words and their surround-

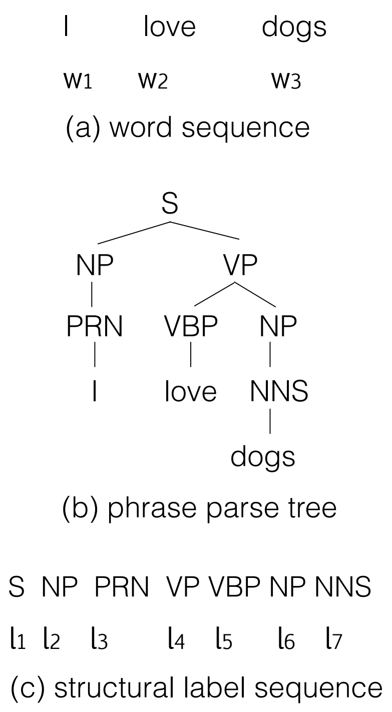

Figure 3: An example of an input sentence (a), its parse tree (b), and the parse tree's sequential form (c).

ings, but also structural context in the parse tree. In the rest of this section, we use English sentences as examples to explain our methods.

3.1 Syntax Representation

To obtain the structural context of a word in its parse tree, ideally the model should not only capture and remember the whole parse tree structure, but also discriminate the contexts of any two different words. However, considering the lack of efficient way to directly model structural information, an alternative way is to linearize the phrase parse tree into a sequence of structural labels and learn the structural context through the sequence. For example, Figure 3(c) shows the structural label sequence of Figure 3(b) in a simple way following a depth-first traversal order. Note that linearizing a parse tree in a depth-first traversal order into a sequence of structural labels has also been widely adopted in recent advances in neural syntactic parsing (Vinyals et al., 2015; Choe and Charniak, 2016), suggesting that the linearized sequence can be viewed as an alternative to its tree structure.[2]

[2]We have also tried to include the ending brackets in the structural label sequence, as what (Vinyals et al., 2015; Choe and Charniak, 2016) do. However, the performance gap is very small by adding the ending brackets or not.

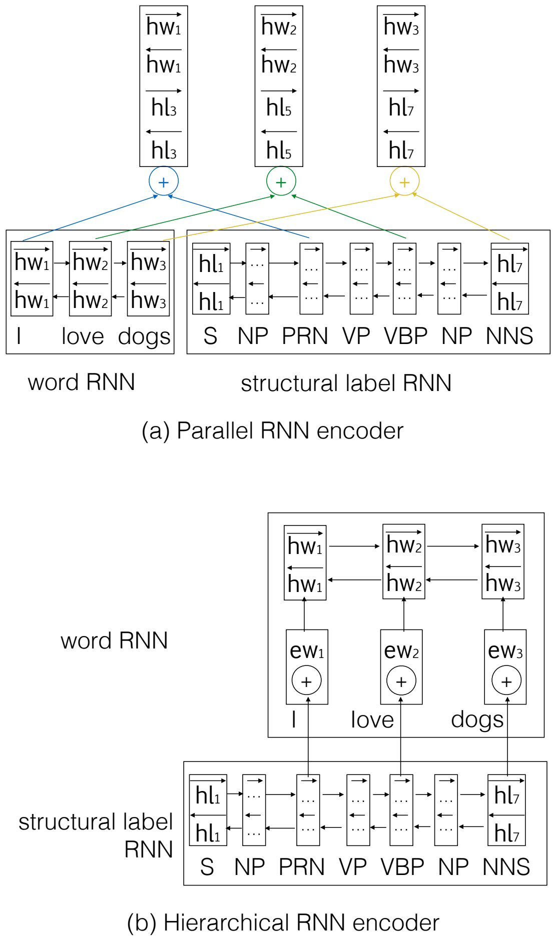

Figure 4: The graphical illustration of the *Parallel RNN* encoder (a) and the *Hierarchical RNN* encoder (b). Here, $\overrightarrow{hw_j}$ and $\overleftarrow{hw_j}$ are the forward and backward hidden states for word w_j, $\overrightarrow{hl_i}$ and $\overleftarrow{hl_i}$ are for structural label l_i, ew_j is the word embedding for word w_j, and $\bigoplus$ is for concatenation operator.

There is no doubt that the structural label sequence is much longer than its word sequence. In order to obtain the structural label annotation vector for w_i in word sequence, we simply look for w_i's part-of-speech (POS) tag in the label sequence and view the tag's annotation vector as w_i's label annotation vector. This is because w_i's POS tag location can also represent w_i's location in the parse tree. For example, in Figure 3, word w_1 in (a) maps to l_3 in (c) since l_3 is the POS tag of w_1. Likewise, w_2 maps to l_5 and w_3 to l_7. That is to say, we use l_3's learned annotation vector as w_1's label annotation vector.

3.2 RNN Encoders with Source Syntax

In the next, we first propose two different encoders to augment word annotation vector with its corresponding label annotation vector, each of which consists of two RNNs [3]: in one encoder, the two RNNs work independently (i.e., *Parallel RNN Encoder*) while in another encoder the two RNNs work in a hierarchical way (i.e., *Hierarchical RNN Encoder*). The difference between the two encoders lies in how the two RNNs interact. Then, we propose the third encoder with a single RNN, which learns word and label annotation vectors stitchingly (i.e., *Mixed RNN Encoder*). Since any of the above three approaches focuses only on the encoder as to generate source annotation vectors along with structural information, we keep the rest part of the NMT models unchanged.

Parallel RNN Encoder Figure 4 (a) illustrates our *Parallel RNN* encoder, which includes two parallel RNNs: i.e., a word RNN and a structural label RNN. On the one hand, the word RNN, as in conventional NMT models, takes a word sequence as input and output a word annotation vector for each word. On the other hand, the structural label RNN takes the structural label sequence of the word sequence as input and obtains a label annotation vector for each label. Besides, we concatenate each word's word annotation vector and its POS tag's label annotation vector as the final annotation vector for the word. For example, the final annotation vector for word *love* in Figure 4 (a) is $[\overrightarrow{hw_2};\overleftarrow{hw_2};\overrightarrow{hl_5};\overleftarrow{hl_5}]$, where the first two subitems $[\overrightarrow{hw_2};\overleftarrow{hw_2}]$ are the word annotation vector and the rest two subitems $[\overrightarrow{hl_5};\overleftarrow{hl_5}]$ are its POS tag *VBP*'s label annotation vector.

Hierarchical RNN Encoder Partially inspired by the model architecture of GNMT (Wu et al., 2016) which consists of multiple layers of LSTM RNNs, we propose a two-layer model architecture in which the lower layer is the structural label RNN while the upper layer is the word RNN, as shown in Figure 4 (b). We put the word RNN in the upper layer because each item in the word sequence can map into an item in the structural label sequence, while this does not hold if the order of the two RNNs is reversed. As shown in Figure 4 (b), for example, the POS tag *VBP*'s label annotation vector $[\overrightarrow{hl_5},\overleftarrow{hl_5}]$ is concatenated with word *love*'s word embedding ew_2 to feed as the input to the word RNN.

[3]Hereafter, we simplify bi-RNN as RNN.

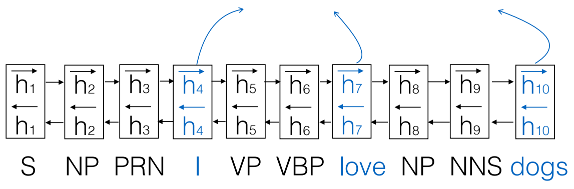

Figure 5: The graphical illustration of the *Mixed RNN* encoder. Here, $\overrightarrow{h_j}$ and $\overleftarrow{h_j}$ are the forward and backward hidden annotation vectors for j-th item, which can be either a word or a structural label.

Mixed RNN Encoder Figure 5 presents our *Mixed RNN* encoder. Similarly, the sequence of input is the linearization of its parse tree (as in Figure 3 (b)) following a depth-first traversal order, but being mixed with both words and structural labels in a stitching way. It shows that the RNN learns annotation vectors for both the words and the structural labels, though only the annotation vectors of words are further fed to decoding (e.g., $([\overrightarrow{h_4},\overleftarrow{h_4}],[\overrightarrow{h_7},\overleftarrow{h_7}],[\overrightarrow{h_{10}},\overleftarrow{h_{10}}])$). Even though the annotation vectors of structural labels are not directly fed forward for decoding, the error signal is back propagated along the word sequence and allows the annotation vectors of structural labels being updated accordingly.

3.3 Comparison of RNN Encoders with Source Syntax

Though all the three encoders model both word sequence and structural label sequence, the differences lie in their respective model architecture with respect to the degree of coupling the two sequences:

- In the *Parallel RNN* encoder, the word RNN and structural label RNN work in a parallel way. That is to say, the error signal back propagated from the word sequence would not affect the structural label RNN, and vice versa. In contrast, in the *Hierarchical RNN* encoder, the error signal back propagated from the word sequence has a direct impact on the structural label annotation vectors, and thus on the structural label embeddings. Finally, the *Mixed RNN* encoder ties the structural label sequence and word sequence together in the closest way. Therefore, the degrees of coupling the word and structural label sequences in these three encoders are like this: *Mixed RNN* encoder $>$ *Hierarchical RNN* encoder $>$ *Parallel RNN* encoder.

- Figure 4 and Figure 5 suggest that the *Mixed RNN* encoder is the simplest. Moreover, comparing to conventional NMT encoders, the difference lies only in the length of the input sequence. Statistics on our training data reveal that the *Mixed RNN* encoder approximately triples the input sequence length compared to conventional NMT encoders.

4 Experimentation

We have presented our approaches to incorporating the source syntax into NMT encoders. In this section, we evaluate their effectiveness on Chinese-to-English translation.

4.1 Experimental Settings

Our training data for the translation task consists of 1.25M sentence pairs extracted from LDC corpora, with 27.9M Chinese words and 34.5M English words respectively.[4] We choose NIST MT 06 dataset (1664 sentence pairs) as our development set, and NIST MT 02, 03, 04, and 05 datasets (878, 919, 1788 and 1082 sentence pairs, respectively) as our test sets.[5] To get the source syntax for sentences on the source-side, we parse the Chinese sentences with Berkeley Parser [6] (Petrov and Klein, 2007) trained on Chinese TreeBank 7.0 (Xue et al., 2005). We use the case insensitive 4-gram NIST BLEU score (Papineni et al., 2002) for the translation task.

For efficient training of neural networks, we limit the maximum sentence length on both source and target sides to 50. We also limit both the source and target vocabularies to the most frequent 16K words in Chinese and English, covering approximately 95.8% and 98.2% of the two corpora respectively. All the out-of-vocabulary words are mapped to a special token *UNK*. Besides, the word embedding dimension is 620 and the size of a hidden layer is 1000. All the other settings are the same as in Bahdanau et al.(2015).

The inventory of structural labels includes 16 phrase labels and 32 POS tags. In both our *Parallel RNN* encoder and *Hierarchical RNN* encoder, we set the embedding dimension of these labels as 100 and the size of a hidden layer as 100. Besides, the maximum structural label sequence length is set to 100. In our *Mixed RNN* encoder, since we only have one input sequence, we equally treat the structural labels and words (i.e., a structural label is also initialized with 620 dimension embedding). Compared to the baseline NMT model, the only different setting is that we increase the maximum sentence length on source-side from 50 to 150.

We compare our method with two state-of-the-art models of SMT and NMT:

- cdec (Dyer et al., 2010): an open source hierarchical phrase-based SMT system (Chiang, 2007) with default configuration and a 4-gram language model trained on the target portion of the training data.[7]

- RNNSearch: a re-implementation of the attentional NMT system (Bahdanau et al., 2015) with slight changes taken from dl4mt tutorial.[8] For the activation function f of an RNN, RNNSearch uses the gated recurrent unit (GRU) recently proposed by (Cho et al., 2014b). It incorporates dropout (Hinton et al., 2012) on the output layer and improves the attention model by feeding the lastly generated word. We use AdaDelta (Zeiler, 2012) to optimize model parameters in training with the mini-batch size of 80. For translation, a beam search with size 10 is employed.

4.2 Experiment Results

Table 1 shows the translation performances measured in BLEU score. Clearly, all the proposed NMT models with source syntax improve the translation accuracy over all test sets, although there exist considerable differences among different variants.

Parameters The three proposed models introduce new parameters in different ways. As a baseline model, RNNSearch has 60.6M parameters. Due to the infrastructure similarity, the *Parallel RNN* system and the *Hierarchical RNN* system introduce the similar size of additional parameters, resulting from the RNN model for structural label sequences (about 0.1M parameters) and catering either the augmented annotation vectors (as shown

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

[5]`http://www.itl.nist.gov/iad/mig/tests/mt/`

[6]`https://github.com/slavpetrov/berkeleyparser`

[7]`https://github.com/redpony/cdec`

[8]`https://github.com/nyu-dl/dl4mt-tutorial`

#	System	#Params	Time	MT06	MT02	MT03	MT04	MT05	All
1	cdec	-	-	33.4	34.8	33.0	35.7	32.1	34.2
2	RNNSearch	60.6M	153m	34.0	36.9	33.7	37.0	34.1	35.6
3	Parallel RNN	+1.1M	+9m	34.8†	**37.8**‡	34.2	38.3‡	34.6	36.6‡
4	Hierarchical RNN	+1.2M	+9m	35.2‡	37.2	34.7†	38.7‡	34.7†	36.7‡
5	Mixed RNN	+0	+40m	**35.6**‡	37.7†	**34.9**‡	**38.6**‡	**35.5**‡	**37.0**‡

Table 1: Evaluation of the translation performance. † and ‡: significant over RNNSearch at 0.05/0.01, tested by bootstrap resampling (Koehn, 2004). "+" is the additional number of parameters or training time on the top of the baseline system RNNSearch. Column *Time* indicates the training time in minutes per epoch for different NMT models

in Figure 4 (a)) or the augmented word embeddings (as shown in Figure 4 (b)) (the remain parameters). It is not surprising that the *Mixed RNN* system does not require any additional parameters since though the input sequence becomes longer, we keep the vocabulary size unchanged, resulting in no additional parameters.

Speed Introducing the source syntax slightly slows down the training speed. When running on a single GPU GeForce GTX 1080, the baseline model speeds 153 minutes per epoch with 14K updates while the proposed structural label RNNs in both *Parallel RNN* and *Hierarchical RNN* systems only increases the training time by about 6% (thanks to the small size of structural label embeddings and annotation vectors), and the *Mixed RNN* system spends 26% more training time to cater the triple sized input sequence.

Comparison with the baseline NMT model (RNNSearch) While all the three proposed NMT models outperform RNNSearch, the *Parallel RNN* system and the *Hierarchical RNN* system achieve similar accuracy (e.g., 36.6 *v.s.* 36.7). Besides, the *Mixed RNN* system achieves the best accuracy overall test sets with the only exception of NIST MT 02. Over all test sets, it outperforms RNNSearch by 1.4 BLEU points and outperforms the other two improved NMT models by 0.3∼0.4 BLEU points, suggesting the benefits of high degree of coupling the word sequence and the structural label sequence. This is very encouraging since the *Mixed RNN* encoder is the simplest, without introducing new parameters and with only slight additional training time.

Comparison with the SMT model (cdec) Table 1 also shows that all NMT systems outperform the SMT system. This is very consistent with other studies on Chinese-to-English translation (Mi et al., 2016; Tu et al., 2017b; Wang et al., 2017).

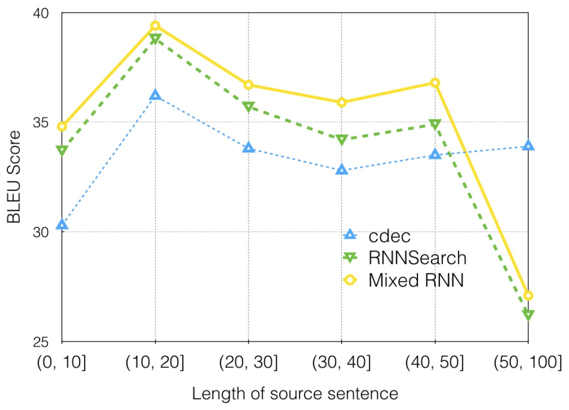

Figure 6: Performance of the generated translations with respect to the lengths of the input sentences.

5 Analysis

As the proposed *Mixed RNN* system achieves the best performance, we further look at the RNNSearch system and the *Mixed RNN* system to explore more on how syntactic information helps in translation.

5.1 Effects on Long Sentences

Following Bahdanau et al. (2015), we group sentences of similar lengths together and compute BLEU scores. Figure 6 presents the BLEU scores over different lengths of input sentences. It shows that *Mixed RNN* system outperforms RNNSearch over sentences with all different lengths. It also shows that the performance drops substantially when the length of input sentences increases. This performance trend over the length is consistent with the findings in (Cho et al., 2014a; Tu et al., 2016, 2017a). We also observe that the NMT sys-

System	AER
RNNSearch	50.1
Mixed RNN	47.9

Table 2: Evaluation of alignment quality. The lower the score, the better the alignment quality.

tems perform surprisingly bad on sentences over 50 in length, especially compared to the performance of SMT system (i.e., cdec). We think that the bad behavior of NMT systems towards long sentences (e.g., length of 50) is due to the following two reasons: (1) the maximum source sentence length limit is set as 50 in training, [9] making the learned models not ready to translate sentences over the maximum length limit; (2) NMT systems tend to stop early for long input sentences.

5.2 Analysis on Word Alignment

Due to the capability of carrying syntactic information in source annotation vectors, we conjecture that our model with source syntax is also beneficial for alignment. To test this hypothesis, we carry out experiments of the word alignment task on the evaluation dataset from Liu and Sun (2015), which contains 900 manually aligned Chinese-English sentence pairs. We force the decoder to output reference translations, as to get automatic alignments between input sentences and their reference translations. To evaluate alignment performance, we report the alignment error rate (AER) (Och and Ney, 2003) in Table 2.

Table 2 shows that source syntax information improves the attention model as expected by maintaining an annotation vector summarizing structural information on each source word.

5.3 Analysis on Phrase Alignment

The above subsection examines the alignment performance at the word level. In this subsection, we turn to phrase alignment analysis by moving from word unit to phrase unit. Given a source phrase *XP*, we use word alignments to examine if the phrase is translated continuously (**Cont.**), or discontinuously (**Dis.**), or if it is not translated at all (**Un.**).

There are some phrases, such as noun phrases (NPs), prepositional phrases (PPs) that we usually expect to have a continuous translation. With respect to several such types of phrases, Table 3 shows how these phrases are translated. From the table, we see that translations of RNNSearch system do not respect source syntax very well. For example, in RNNSearch translations, 57.3%, 33.6%, and 9.1% of PPs are translated continuously, discontinuously, and untranslated, respectively. Fortunately, our *Mixed RNN* system is able to have more continuous translation for those phrases. Table 3 also suggests that there is still much room for NMT to show more respect to syntax.

[9]Though the maximum source length limit in *Mixed RNN* system is set to 150, it approximately contains 50 words in maximum.

System	XP	Cont.	Dis.	Un.
RNNSearch	PP	57.3	33.6	9.1
	NP	59.8	25.5	14.7
	CP	47.3	44.6	8.1
	QP	54.0	22.2	23.8
	ALL	58.1	27.1	14.8
Mixed RNN	PP	63.3	27.5	9.2
	NP	63.1	23.1	13.8
	CP	54.5	36.6	8.9
	QP	56.2	19.7	24.1
	ALL	60.4	25.0	14.6

Table 3: Percentages (%) of syntactic phrases in our test sets being translated continuously, discontinuously, or not being translated. Here PP is for prepositional phrase, NP for noun phrase, CP for clause headed by a complementizer, QP for quainter phrase.

5.4 Analysis on Over Translation

To estimate the over translation generated by NMT, we propose *ratio of over translation (ROT)*:

$$ROT = \frac{\sum_{w_i} t(w_i)}{|w|} \tag{1}$$

where $|w|$ is the number of words in consideration, $t(w_i)$ is the times of over translation for word w_i. Given a word w and its translation $e = e_1 e_2 \dots e_n$, we have:

$$t(w) = |e| - |uniq(e)| \tag{2}$$

where $|e|$ is the number of words in w's translation e, while $|uniq(e)|$ is the number of unique words in e. For example, if a source word 香港*/xiangkang* is translated as *hong kong hong kong*, we say it being over translated *2* times.

Table 4 presents ROT grouped by some typical POS tags. It is not surprising that RNNSearch system has high ROT with respect to POS tags of NR

System	POS	ROT (%)
RNNSearch	NR	15.7
	CD	7.4
	DT	4.9
	NN	8.0
	ALL	5.5
Mixed RNN	NR	12.3
	CD	5.1
	DT	2.4
	NN	6.8
	ALL	4.5

Table 4: Ratio of over translation (ROT) on test sets. Here NR is for proper noun, CD for cardinal number, DT for determiner, and NN for nouns except proper nouns and temporal nouns.

System	POS	non-UNK	UNK	Un.
RNNSearch	NN	27.2	40.4	32.4
	NR	22.9	58.5	18.6
	VV	34.5	32.9	32.7
	CD	10.7	63.4	25.9
	ALL	27.2	40.4	32.4
Mixed RNN	NN	24.8	41.4	33.8
	NR	17.0	64.5	18.6
	VV	33.6	34.0	32.3
	CD	9.6	68.7	21.7
	ALL	23.9	47.5	28.7

Table 5: Percentages (%) of rare words in our test sets being translated into a *non-UNK* word (**non-UNK**), *UNK* (**UNK**), or if it is not translated at all (**Un.**).

(proper noun) and CD (cardinal number): this is due to the fact that the two POS tags include high percentage of unknown words which tend to be translated multiple times in translation. Words of DT (determiner) are another source of over translation since they are usually translated to multiple *the* in English. It also shows that by introducing source syntax, *Mixed RNN* system alleviates the over translation issue by 18%: ROT drops from 5.5% to 4.5%.

5.5 Analysis on Rare Word Translation

We analyze the translation of source-side rare words that are mapped to a special token *UNK*. Given a rare word *w*, we examine if it is translated into a *non-UNK* word (**non-UNK**), *UNK* (**UNK**), or if it is not translated at all (**Un.**).

Table 5 shows how source-side rare words are translated. The four POS tags listed in the table account for about 90% of all rare words in the test sets. It shows that in *Mixed RNN* system is more likely to translate source-side rare words into *UNK* on the target side. This is reasonable since the source side rare words tends to be translated into rare words in the target side. Moreover, it is hard to obtain its correct non-*UNK* translation when a source-side rare word is replaced as *UNK*.

Note that our approach is compatible with with approaches of open vocabulary. Taking the subword approach (Sennrich et al., 2016) as an example, for a word on the source side which is divided into several subword units, we can synthesize sub-POS nodes that cover these units. For example, if *misunderstand/VB* is divided into units of *mis* and *understand*, we construct substructure *(VB (VB-F mis) (VB-I understand))*.

6 Related Work

While there has been substantial work on linguistically motivated SMT, approaches that leverage syntax for NMT start to shed light very recently. Generally speaking, NMT can provide a flexible mechanism for adding linguistic knowledge, thanks to its strong capability of automatically learning feature representations.

Eriguchi et al. (2016) propose a tree-to-sequence model that learns annotation vectors not only for terminal words, but also for non-terminal nodes. They also allow the attention model to align target words to non-terminal nodes. Our approach is similar to theirs by using source-side phrase parse tree. However, our *Mixed RNN* system, for example, incorporates syntax information by learning annotation vectors of syntactic labels and words stitchingly, but is still a sequence-to-sequence model, with no extra parameters and with less increased training time.

Sennrich and Haddow (2016) define a few linguistically motivated features that are attached to each individual words. Their features include lemmas, subword tags, POS tags, dependency labels, etc. They concatenate feature embeddings with word embeddings and feed the concatenated embeddings into the NMT encoder. On the contrast, we do not specify any feature, but let the model implicitly learn useful information from the structural label sequence.

Shi et al. (2016) design a few experiments to investigate if the NMT system without external linguistic input is capable of learning syntactic information on the source-side as a by-product of training. However, their work is not focusing on improving NMT with linguistic input. Moreover, we analyze what syntax is disrespected in translation from several new perspectives.

García-Martínez et al. (2016) generalize NMT outputs as lemmas and morphological factors in order to alleviate the issues of large vocabulary and out-of-vocabulary word translation. The lemmas and corresponding factors are then used to generate final words in target language. Though they use linguistic input on the target side, they are limited to the word level features. Phrase level, or even sentence level linguistic features are harder to obtain for a generation task such as machine translation, since this would require incremental parsing of the hypotheses at test time.

7 Conclusion

In this paper, we have investigated whether and how source syntax can explicitly help NMT to improve its translation accuracy.

To obtain syntactic knowledge, we linearize a parse tree into a structural label sequence and let the model automatically learn useful information through it. Specifically, we have described three different models to capture the syntax knowledge, i.e., *Parallel RNN*, *Hierarchical RNN*, and *Mixed RNN*. Experimentation on Chinese-to-English translation shows that all proposed models yield improvements over a state-of-the-art baseline NMT system. It is also interesting to note that the simplest model (i.e., *Mixed RNN*) achieves the best performance, resulting in obtaining significant improvements of 1.4 BLEU points on NIST MT 02 to 05.

In this paper, we have also analyzed the translation behavior of our improved system against the state-of-the-art NMT baseline system from several perspectives. Our analysis shows that there is still much room for NMT translation to be consistent with source syntax. In our future work, we expect several developments that will shed more light on utilizing source syntax, e.g., designing novel syntactic features (e.g., features showing the syntactic role that a word is playing) for NMT, and employing the source syntax to constrain and guild the attention models.

Acknowledgments

The authors would like to thank three anonymous reviewers for providing helpful comments, and also acknowledge Xing Wang, Xiangyu Duan, Zhengxian Gong for useful discussions. This work was supported by National Natural Science Foundation of China (Grant No. 61525205, 61331011, 61401295).

References

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

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics* 33(2):201–228.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machinetranslation: Encoder-decoder approaches. In *Proceedings of SSST 2014*. pages 103–111.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of EMNLP 2014*. pages 1724–1734.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Proceedings of EMNLP 2016*. pages 2331–2336.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of ACL 2010 System Demonstrations*. pages 7–12.

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of ACL 2016*. pages 823–833.

Mercedes García-Martínez, Loic Barrault, and Fethi Bougares. 2016. Factored neural machine translation. In *arXiv:1609.04621*.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. In *arXiv:1207.0580*.

Sébastien Jean, Orhan Firat, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. Montreal neural machine translation systems for wmt'15. In *Proceedings of WMT 2015*. pages 134–140.

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

Junhui Li, Philip Resnik, and Hal Daumé III. 2013. Modeling syntactic and semantic structures in hierarchical phrase-based translation. In *Proceedings of HLT-NAACL 2013*. pages 540–549.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of ACL-COLING 2006*. pages 609–616.

Yang Liu and Maosong Sun. 2015. Contrastive unsupervised word alignment with non-local features. In *Proceedings of AAAI 2015*. pages 857–868.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of IWSLT 2015*. pages 76–79.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP 2015*. pages 1412–1421.

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proceedings of ACL-HLT 2008*. pages 1003–1011.

Haitao Mi, Zhiguo Wang, and Abe Ittycheriah. 2016. Supervised attentions for neural machine translation. In *Proceedings of EMNLP 2016*. pages 2283–2288.

Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics* 29(1):19–51.

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*. pages 311–318.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HLT-NAACL 2007*. pages 404–411.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation*. pages 83–91.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL 2016*. pages 1715–1725.

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

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of EMNLP 2016*. pages 1526–1534.

Zhaopeng Tu, Yang Liu, Zhengdong Lu, Xiaohua Liu, and Hang Li. 2017a. Context gates for neural machine translation. *Transactions of the Association for Computational Linguistics* 5:87–99.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017b. Neural machine translation with reconstruction. In *Proceedings of AAAI 2017*. pages 3097–3103.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of ACL 2016*. pages 76–85.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Proceedings of NIPS 2015*.

Xing Wang, Zhengdong Lu, Zhaopeng Tu, Hang Li, Deyi Xiong, and Min Zhang. 2017. Neural machine translation advised by statistical machine translation. In *Proceedings of AAAI 2017*. pages 3330–3336.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. In *arXiv preprint arXiv:1609.08144*.

Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering* 11(2):207–238.

Matthew D. Zeiler. 2012. ADADELTA: An adaptive learning rate method. In *arXiv:1212.5701*.