Refining Source Representations with Relation Networks for Neural Machine Translation

Wen Zhang[1,2] **Jiawei Hu**[1,2] **Yang Feng**[1,2*] **Qun Liu**[3,1]

[1]Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, CAS
[2]University of Chinese Academy of Sciences
{zhangwen,hujiawei,fengyang}@ict.ac.cn
[3]ADAPT Centre, School of Computing, Dublin City University
qun.liu@dcu.ie

Abstract

Although neural machine translation with the encoder-decoder framework has achieved great success recently, it still suffers drawbacks of forgetting distant information, which is an inherent disadvantage of recurrent neural network structure, and disregarding relationship between source words during encoding step. Whereas in practice, the former information and relationship are often useful in current step. We target on solving these problems and thus introduce relation networks to learn better representations of the source. The relation networks are able to facilitate memorization capability of recurrent neural network via associating source words with each other, this would also help retain their relationships. Then the source representations and all the relations are fed into the attention component together while decoding, with the main encoder-decoder framework unchanged. Experiments on several datasets show that our method can improve the translation performance significantly over the conventional encoder-decoder model and even outperform the approach involving supervised syntactic knowledge.

1 Introduction

In recent years, Neural Machine Translation (NMT) (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Bahdanau et al., 2015) has achieved great success in some language pairs, rivalling the state-of-the-art Statistical Machine Translation (SMT). The Recurrent Neural Network (RNN) encoder-decoder architecture is widely used framework for NMT, the principle behind which is that: encoding the meaning of the input bidirectionally into a concept space via RNNs and decoding into target words with RNNs based on this encoding (Sutskever et al., 2014; Bahdanau et al., 2015). This means that encoding principle leads to a deeper understanding and learning of the translation rules, and hence better translation than conventional SMT that considers only surface forms, e.g., words and phrases.

The RNNs with gating, such as Gated Recurrent Unit (GRU) (Cho et al., 2014) or Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), are designed to memorize useful history information and meanwhile forget irrelative information. Together with attention technique which makes the decoding process only focus on the most related source words, the RNN encoder-decoder framework is expected to be able to handle long sequences and consider the globally related information. However, the practical situation is that RNNs tend to forget old history information, especially the far older one. Sometimes the older information is indispensable for generating proper translation, e.g., for the source sentence "take the heavy box away", when translating "away", "take" should be considered together. In addition, it has been proven that using phrases rather than words in SMT (Koehn et al., 2003) brings performance improvement, while in NMT the attention is only modeled in the unit of words. In the same sense, improvement is expected if attention is operated on more words rather than one.

Moreover, NMT produces the representation for the source by running through the source words sequentially with a bidirectional RNN (Schuster and Paliwal, 1997), so it only employs word order information and ignores the relation between words. Although some researchers have demonstrated that

*Corresponding author.

Proceedings of the 27th International Conference on Computational Linguistics, pages 1292–1303
Santa Fe, New Mexico, USA, August 20-26, 2018.

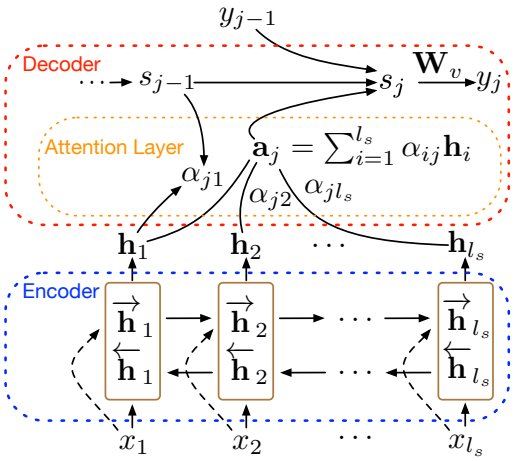


Figure 1: The architecture of attention-based NMT

NMT is able to capture certain syntactic phenomena (e.g. subject-verb agreement) without external syntactic information (Linzen et al., 2016; Shi et al., 2016), there are some other works which has shown their superior performance by modeling word relationship explicitly. However, these works usually need to introduce external syntactic knowledge or connect words according to their relations in the syntactic structure (Sennrich et al., 2016; Bastings et al., 2017; Aharoni and Goldberg, 2017; Li et al., 2017).

In this paper, we present a method to refine the NMT based on the above two points. The main idea is to learn relationship between the source word pairs. Corresponding to the first point, our method employs Convolutional Neural Networks (CNNs) to collect local information around one word and relates each word with its neighbors, which ensures the subsequent operations are performed in the unit of multiple words. As for the second point, Relation Network (RN) (Santoro et al., 2017) is introduced to establish pairwise relationship between words, meanwhile, there's no need to attain external input of syntactic knowledge. In this way, our model can memorize all words ahead and behind via additional connection between words no matter how distant they are. In the RNs, the representations of the source words produced by RNNs are taken as objects and the relationships between them are reasoned.

Specifically, our method introduces a RN component between the encoder and the attention layer in the RNN encoder-decoder framework (Sutskever et al., 2014; Bahdanau et al., 2015). The RN component is composed of three layers: first, the CNN layer slides window along the output of the encoder to capture information among multiple words around one word, then the graph propagation layer constructs a fully connected graph with the information of one window as one node and transfers messages along the edges, so that each node can collect the information from all other nodes, and last the multi-layer perceptron layer transforms the information of each node to the form which is suitable for the attention component to use. We performed experiments on several datasets and got significant improvements over vanilla NMT and SMT systems. Besides, our model significantly outperforms two other models, which introduced latent variables to capture the implicit semantics and employed explicitly external syntactic knowledge respectively.

2 Background

As the main idea of our method is to introduce relation networks into the attention-based NMT (Bahdanau et al., 2015) to learn word relationship and keep all source words in memory, in this section we will briefly describe the baseline model – the attention-based NMT first and the technique used in this paper – relation networks.

2.1 Attention-based NMT

The attention-based NMT follows the encoder-decoder framework, with an additional attention module. It works on the assumption that the source sentence and the target translation share a common continuous

space. It first encodes the source sentence into a continuous space and then performs decoding based on this space, meanwhile, employing attention to indicate the relevance of each source word to the current translation. Figure 1 shows the architecture of the attention-based NMT (Bahdanau et al., 2015), which is composed of three components: the encoder, the attention layer and the decoder.

The Encoder The encoder uses a pair of GRUs to run through source words bidirectionally to get two sequences of hidden states, which are concatenated to produce corresponding hidden state for the i-th source word

$$\overrightarrow{\mathbf{h}}_i = \overrightarrow{\mathbf{GRU}}\left(x_i, \overrightarrow{\mathbf{h}}_{i-1}\right); \qquad \overleftarrow{\mathbf{h}}_i = \overleftarrow{\mathbf{GRU}}\left(x_i, \overleftarrow{\mathbf{h}}_{i+1}\right); \qquad \mathbf{h}_i = \left[\overrightarrow{\mathbf{h}}_i; \overleftarrow{\mathbf{h}}_i\right] \tag{1}$$

The Attention Layer The attention layer aims to extract the source information (called attention) which is highly related to the generation of the current target word. To get the attention of the j-th decoding step, the correlation degree between current target word y_j and $\mathbf{h}_i$ is first evaluated as

$$e_{ij} = \mathbf{v}_a^T \tanh(\mathbf{W}_a \boldsymbol{s}_{j-1} + \mathbf{U}_a \mathbf{h}_i) \tag{2}$$

Then, for the j-th decoding step, the correlation degree is normalized over the whole source sequence, all source hidden states are added weightedly according to the normalized correlation degree to obtain the attention $\mathbf{a}_j$

$$\alpha_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{i'=1}^{l_s} \exp\left(e_{i'j}\right)}; \qquad \mathbf{a}_j = \sum\nolimits_{i=1}^{l_s} \alpha_{ij} \mathbf{h}_i \tag{3}$$

The Decoder The decoder first employs a variant of GRU to roll the target information according to previous target word y_{j-1}, previous hidden state $\boldsymbol{s}_{j-1}$ and the attention $\mathbf{a}_j$. The details are described in Bahdanau et al. (2015). The current target hidden state $\boldsymbol{s}_j$ is calculated by

$$\boldsymbol{s}_j = g(y_{j-1}, \boldsymbol{s}_{j-1}, \mathbf{a}_j) \tag{4}$$

After that, the decoder gives a probability distribution over all the words in the target vocabulary and selects the target word with the highest probability as the output of the current step

$$p(y_j | \mathbf{y}_{<j}, \mathbf{x}) \propto \exp(f(\boldsymbol{s}_j, y_{j-1}, \mathbf{a}_j) \cdot \mathbf{W}_{\boldsymbol{v}}) \tag{5}$$

where f stands for a linear transformation and $\mathbf{W}_{\boldsymbol{v}}$ is a weight matrix.

2.2 Relation Networks

A relation network (RN) is a neural network with a structure integrated for relational reasoning. The RN is designed to constrain the functional form of a neural network so that it can capture the core common properties of relational reasoning. Hence its capability of computing relations is inherent without needing to be learned specially.

Formally, given a set of input "objects" denoted as $\mathrm{O} = \{o_1, o_2, \cdots, o_n\}$, RN can be formed as a composition function of objects (Santoro et al., 2017), represented as

$$\mathrm{RN}(\mathrm{O}) = f_\phi\left(\sum\nolimits_{i,j} g_\theta\left(o_i, o_j\right)\right) \tag{6}$$

where o_i is the i-th object, and f_ϕ and g_θ are functions used to calculate relations. Multi-layer perceptrons are often used for f_ϕ and g_θ, as their parameters are learnable synaptic weights, making RNs end-to-end differentiable. Here the role of g_θ is to infer how two objects are related, or whether they are related, and hence the output of g_θ can be treated as "relations".

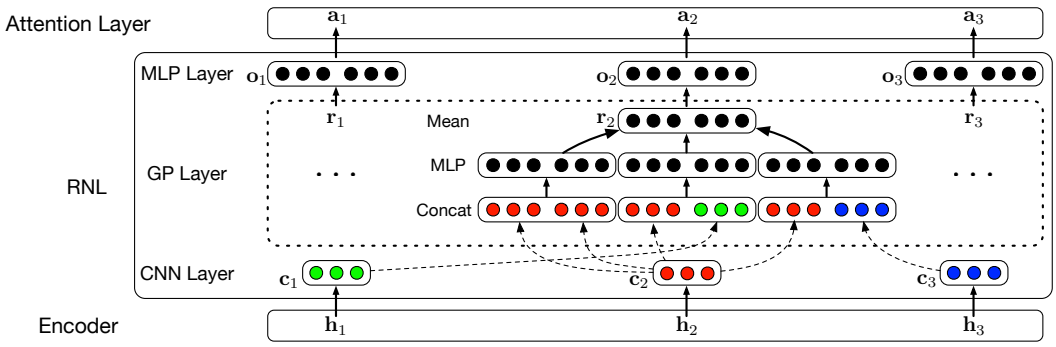


Figure 2: NMT with one RNL. Residual connection is not embodied here. The kernel width of the CNN layer is 3. We take the second word colored in red as an example to show the operations in the RNL, where three colors of green, red and blue indicate the information from the CNN layer of the first, second and third word, respectively.

3 NMT with Relation Networks

In this paper, we introduce a Relation Network Layer (denoted as RNL) on the basis of the attention-based NMT (Bahdanau et al., 2015) and frame it between the encoder and the attention layer. The RNL first employs CNNs to collect information in the unit of multi-words rather than one single word, then takes the outputs of CNNs as objects and makes them fully connected to build a graph propagation layer and associate with each other, finally transforms the acquired representations with word relations via MLP into the form suitable for the attention layer to use. Next, the outputs of the RNL are directly fed into the attention layer, so the RNL can still fit the encoder-decoder framework well. The architecture of our RNL is shown in Figure 2. Briefly, the RNL is composed of three components: the CNN layer, the Graph Propagation (GP) layer and the Multi-layer Perceptron (MLP) layer.

The CNN Layer CNNs are used to collect local information around one word. In this way, not only the information of a single word but their neighbors are considered. The number of neighbors to be considered depends on the kernel width k but can also vary by stacking several convolution layers, e.g., stacking 2 convolution layers with the kernel width $k = 3$ can collect information from 5 words at the same time.

In the CNN layer, the input is the hidden states produced by the bidirectional GRUs (Bi-GRUs), denoted as $\mathbf{h} = \{\mathbf{h}_1, ..., \mathbf{h}_i, ..., \mathbf{h}_{l_s}\}$, so each source word is represented by its hidden state. A filter is applied to convolute over a window of k words to get the convolutional representation. Given the i-th source word and its hidden state $\mathbf{h}_i \in \mathbb{R}^d$, the hidden states covered by the window with the width of k are concatenated and then are fed to the filter where we denote the concatenated vector as $\mathbf{h}_i^k = \left[\mathbf{h}_{i-\lfloor(k-1)/2\rfloor}; \cdots; \mathbf{h}_i; \cdots; \mathbf{h}_{i+\lfloor(k-1)/2\rfloor}\right]$. For the first and last $\lfloor(k-1)/2\rfloor$ words of a sentence, the hidden state $\mathbf{h}_i$ with $i < 1$ or $i > l_s$ are set to zeros (padding). Then the filtering process mentioned above can be formed as

$$\mathbf{c}_i = f\left(\mathbf{W}_{cnn}\mathbf{h}_i^k + \mathbf{b}_{cnn}\right) \tag{7}$$

$\mathbf{W}_{cnn} \in \mathbb{R}^{k\times d}$ is the convolution weights and $\mathbf{b}_{cnn}$ is the bias, where the two together define a linear operation. f is the leaky RELU with the coefficient 0.1 to control the angle of the negative slope. In the RNL, leaky RELU is used as all the nonlinear activation functions. The output of the CNN layer is $\mathbf{c} = \{\mathbf{c}_1, ..., \mathbf{c}_i, ..., \mathbf{c}_{l_s}\}$.

The GP Layer The GP layer is used to learn the relationships between source words. It adopts the outputs of the CNN layer $\mathbf{c} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_{l_s}\}$ as input and formulates the relationships between them into a graph. Here $\mathbf{c}_i$ can be thought as the object mentioned in Section 2.2. In this graph, each input $\mathbf{c}_i$ is taken as a node and has edges connected to all other nodes. Then information flows along the edges and each node receives messages from all its direct neighbors. We call this process graph propagation.

After graph propagation process, another sequence of vectors $\{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{l_s}\}$ is produced. The generation of $\mathbf{r}_i$ can be decomposed into three steps:

- Each input vector $\mathbf{c}_i$ in $\mathbf{c}$ is concatenated with all vectors in $\mathbf{c}$ (including itself) to get a set of vectors $\mathbf{C}_i = \{\mathbf{c}_{i1}, \cdots \mathbf{c}_{ij}, \cdots, \mathbf{c}_{il_s}\}$ where $\mathbf{c}_{ij} = [\mathbf{c}_i; \mathbf{c}_j]$.
- Each $\mathbf{c}_{ij}$ is converted into vector $\mathbf{r}_{ij}$ by a 4-hidden-layers MLP. The conversion with 1-hidden-layer MLP can be represented as
$$\mathbf{r}_{ij} = f\left(\mathbf{W}_{gp}\mathbf{c}_{ij} + \mathbf{b}_{gp}\right) \tag{8}$$
- Average over all the outputs above to get the final representation for the i-th source word
$$\mathbf{r}_i = \frac{1}{l_s}\sum\nolimits_{j=1}^{l_s} \mathbf{r}_{ij} \tag{9}$$

The MLP Layer There are several nonlinear transformations which map the inputs into different vector spaces in the GP layer. In order to reduce computation complexity, the output features size of the nonlinear transformations is set to small. Hence we use another MLP layer to map the feature back into the original space, usually the same as that of $\mathbf{h}_i$ to have more powerful representation. The final state $\mathbf{o}_i$ for the i-th source word after the entire RN layer can be got by another 2-hidden-layers MLP, 1-hidden-layer MLP can be written as

$$\mathbf{o}_i = f(\mathbf{W}_{mlp}\mathbf{r}_i + \mathbf{b}_{mlp}) \tag{10}$$

Residual Stacking technique is used in our method. Concretely, we stack multiple layers inside the encoder and meanwhile apply residual connection for two adjacent layers. Assume h^l_{in} and h^l_{out} are the input and the output of the l-th layer, respectively, then residual connection is conducted to get the final output of the l-th layer in the following two steps. First, the input and the output of the l-th layer are added together:

$$\mathrm{h}^l = \mathrm{h}^l_{in} + \mathrm{h}^l_{out} \tag{11}$$

Next, dense concatenation (Huang et al., 2017) is employed to receives features from all previous layers and the final output of the l-th layer is produced by

$$\mathrm{h}^l_{dc} = \mathbf{W}_{dc}\left[\mathrm{h}^1; \mathrm{h}^2; \cdots; \mathrm{h}^l\right] + \mathbf{b}_{dc} \tag{12}$$

where weight matrix $\mathbf{W}_{dc}$ and bias $\mathbf{b}_{dc}$ are adjusted to map the dense-concatenated vectors into the same feature space as the input. Then h^l_{dc} is fed to the next layer which means $\mathrm{h}^{l+1}_{in} = \mathrm{h}^l_{dc}$.

4 Related Work

Many researchers have worked on learning the relationships of the source words to improve translation performance. One line is to refine source presentations by adding relationships between source words or between source and target words, with the main architecture remaining the RNN encoder-decoder framework. Sennrich et al. (2016) enriched source representations with POS tags, dependency labels and other linguistic features. Bastings et al. (2017) employed graph convolutional networks to model relations of words in dependency trees for the source embeddings to include these relations. These two models both require extra supervised syntax input while our method does not need external knowledge and learn the relationship by its own.

Another line is to change the structure of the neural network. Gehring et al. (2017a) and Gehring et al. (2017b) proposed to substitute the conventional RNN encoder with the CNN encoder in order to train faster. They employed stacked CNNs to capture relationships between source words which can be calculated simultaneously, not like RNNs, the computation of which is constrained by temporal dependencies. The attention scores are also computed based on the output of the CNNs and the decoder is still the RNN-based decoder. Vaswani et al. (2017) is another work to eschew the recurrence. It instead

relied entirely on the attention mechanism to draw the global dependencies between input and output. Su et al. (2018) introduced latent random variables into the decoder of NMT and generated these variables recurrently to capture the global semantic contexts and model strong and complex dependencies among target words at different timesteps.

Our method still follows the RNN encoder-decoder framework, giving full play to the advantages of RNNs, which transfers information through words bidirectionally. In addition, we also employs RNs in our method to connect the source words explicitly, further captures relationships between source words without any external knowledge injection, which enables the model to learn the relationships itself and facilitates easy application.

5 Experiments

In the experiment section, we first compare our system with two baseline systems on a Chinese-English (Zh-En) dataset and the WMT17 English-German (En-De) dataset, then compare our method with a related approach on the WMT16 En-De dataset. Finally, we give some analyses about our method in different aspects.

5.1 Data Preparation

We performed experiments on three datasets:

NIST The training data consisted of 1.25M Zh-En parallel sentence pairs with 25M Chinese tokens and 27M English tokens[1]. We used NIST 2002 test dataset (878 sentences) as the validation set, and another four NIST test datasets as the test datasets: NIST 2003 (MT03), NIST 2004 (MT04), NIST 2005 (MT05) and NIST 2006 (MT06), which contain 919, 1788, 1082 and 1357 sentences respectively.

WMT17 The training data was composed of 5.6M En-De preprocessed parallel sentence pairs [2] with 141M English tokens and 194M German tokens. The test dataset of newstest2014 (3003 sentences) was used as the validation set and the following test datasets were used as the test datasets: newstest2015 (2169 sentences), newstest2016 (2999 sentences) and newstest2017 (3004 sentences). Besides, $8k$ merging operations were performed to learn byte-pair encodings (BPE) (Sennrich et al., 2016) on the target side of the parallel training data.

WMT16 We conducted experiments on WMT16 dataset, the same dataset as the work of Bastings et al. (2017) for comparison. We kept the same settings as those in Bastings et al. (2017): The original dataset consists of 4500966 sentence pairs, with 4173550 left after filtering pairs which contains more than 50 tokens on either side after tokenization. newstest2015 and newstest2016 were used as the validation set and test dataset, respectively. $16k$ BPE merging operations were conducted on the target side of the bilingual training data.

For WMT16 dataset, case-sensitive 4-gram BLEU score (Papineni et al., 2002) was reported by using the *multi-bleu.pl* script. The results on the other two datasets were evaluated with case-insensitive 4-gram BLEU score.

5.2 Systems

Results of five systems on different datasets were reported:

RNNsearch We implemented the attention-based NMT of Bahdanau et al. (2015) by PyTorch framework[3] with the following settings: the length of the sentences on both sides was limited up to 50 tokens with 30K vocabulary, and the source and target word embedding sizes were both set to 512, the size of all hidden units in both encoder and decoder RNNs was also set to 512, and all parameters were initialized by using uniform distribution over $[-0.1, 0.1]$. The mini-batch stochastic gradient descent (SGD) algorithm was employed to train the model with batch size of 80. In addition, the learning rate was adjusted by Adadelta optimizer (Zeiler, 2012) with $\rho = 0.95$ and $\epsilon = 1e$-6. Dropout was applied on the output layer with dropout rate of 0.5. The beam size was set to 10.

[1]We chose LDC2002E18, LDC2003E07, LDC2003E14, Hansard's portion of LDC2004T07, LDC2004T08 and LDC2005T06 from the LDC corpora. There were 1.11M sentence pairs left after filtering.

[2]`http://data.statmt.org/wmt17/translation-task/preprocessed`

[3]`http://pytorch.org`

Systems	MT03	MT04	MT05	MT06	Average
RNNsearch	33.70	36.15	31.81	32.71	33.59
RNNsearch*	37.93	40.53	36.65	35.80	37.73
VRNMT	38.08	41.07	36.82	36.72	38.17
RNMT	**39.24***	**42.01***	**37.79***	**37.81***	**39.21**

Table 1: Performance comparison on NIST datasets. $*$ is used to indicate the improvement over RNNsearch* is statistically significant (Collins et al., 2005) ($p < 0.01$).

Systems	test15	test16	test17	Avg.
RNNsearch	17.3	20.9	16.6	18.3
RNNsearch*	21.4	25.6	20.1	22.4
RNMT	**22.7***	**27.8***	**21.8***	**24.1**

Table 2: Performance comparison on WMT17 En-De datasets.

Systems	test16
BiRNN+GCN	23.9
RNMT	**25.4**

Table 3: Performance comparison with the related work on the WMT16 En-De dataset.

RNNsearch* This system is an improved version of RNNsearch where the decoder employs a conditional GRU layer with attention module, consisting of two GRUs and an attention module for each step[4]. Specifically, Equation 4 is substituted with the following two equations:

$$\tilde{\boldsymbol{s}}_j = \mathbf{GRU}_1(y_{j-1}, \boldsymbol{s}_{j-1}); \qquad \boldsymbol{s}_j = \mathbf{GRU}_2(\mathbf{a}_j, \tilde{\boldsymbol{s}}_j) \tag{13}$$

Besides, for the calculation of attention in Equation 2, $\boldsymbol{s}_{j-1}$ is replaced with $\tilde{\boldsymbol{s}}_{j-1}$. The other components of the system keep the same as RNNsearch. We used the same settings for RNNsearch and RNNsearch*.

VRNMT A novel Variational Recurrent NMT (VRNMT) model, proposed by Su et al. (2018), captures more semantic context and complex dependencies among target words by generating latent random variables recurrently in the NMT decoder.

BiRNN+GCN This is the model presented by Bastings et al. (2017). They incorporated dependency syntactic structure into the bidirectional RNN (BiRNN) encoder of NMT and modeled the relation among the source words by using graph convolutional networks (GCNs).

RNMT Our system was implemented by embedding the RNLs into the Bi-GRUs of RNNsearch*. The overall structure used alternatively stacked GRUs and RNLs, in which the two GRU layers are in opposite direction. Inside the RNL, the GP layer employed a 4-hidden-layers MLP (shown in Equation 8) and the MLP layer contained 2 hidden layers (as in Equation 10). For the Zh-En translation task, two convolution layers with kernel width of 1 and 3 were stacked, the output channel sizes of CNN were 128 and 256 respectively, followed by batch normalization (BN) (Ioffe and Szegedy, 2015) with learnable parameters, and MLP contained 256 units. For the En-De translation task, only one convolution layer was used with kernel width of 3, the output channel size was 96, 128 was adopted as the hidden size of MLP. All of the other settings were the same with those of RNNsearch*.

5.3 Performance Comparison

We compared our system RNMT with the two baseline systems RNNsearch and RNNsearch* both on the NIST Zh-En and the WMT17 En-De translation tasks. As RNMT was implemented on the basis of RNNsearch*, in the strict sense, RNNsearch* is the baseline. From the results shown in Table 1, we can see that RNMT significantly improves translation quality on all test datasets and outperforms RNNsearch* by 1.48 BLEU points averagely on the Zh-En dataset. Besides, comparison between our model to VRNMT shows that proposed simple model stably produces better performance on all test datasets and outperforms VRNMT 1.04 BLEU score on average.

[4] https://github.com/nyu-dl/dl4mt-tutorial/blob/master/docs/cgru.pdf

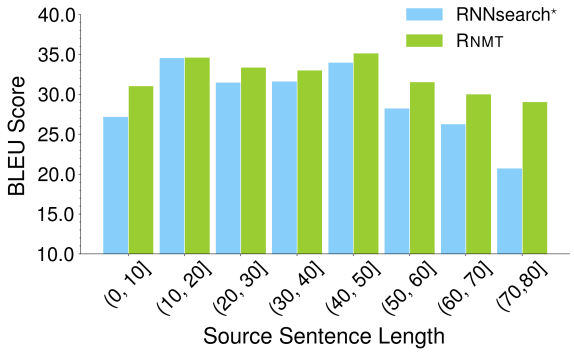


Figure 3: Results on different bins contain sentences of length within corresponding spans.

On the WMT17 En-De dataset, as shown in Table 2, RNMT shows superiority on three test datasets stably, and averagely achieves the gains of 1.7 BLEU points over RNNsearch*, with only 4.1M parameters more. Given the above results, we can conclude that RN can indeed learn the relationships between the source words and these relationships are useful and bring improvement on the translation performance.

We also compared our method with the work of Bastings et al. (2017) which requires the injection of external syntactic knowledge, to see whether the relationships produced by RNs can lead to better translation than the syntax from supervised learning. The results in Table 3 show that our system can achieve an improvement of 1.5 BLEU scores. We believe that the relationships of the source words derived from RNs do not necessarily conform to human cognition, but it can be simultaneously tailored with the other parts of the translation system. In this way, RNs can generate the relationships more suitable for the NMT.

5.4 Impact of Input Length

One motivation of adding RNs is that RNNs tend to forget the distant history which RNs memorize it by explicitly introducing relations between pairs of words. Therefore, we assume that our method suppose to bring greater improvement on relative long sentences, which contains more distant history information than shorter ones that usually forgotten by RNNs. Based on this sense, we split the source sentences in the MT03 test dataset into different bins according to their length and evaluated BLEU scores of the translations from RNNsearch* and RNMT on the different bins, respectively.

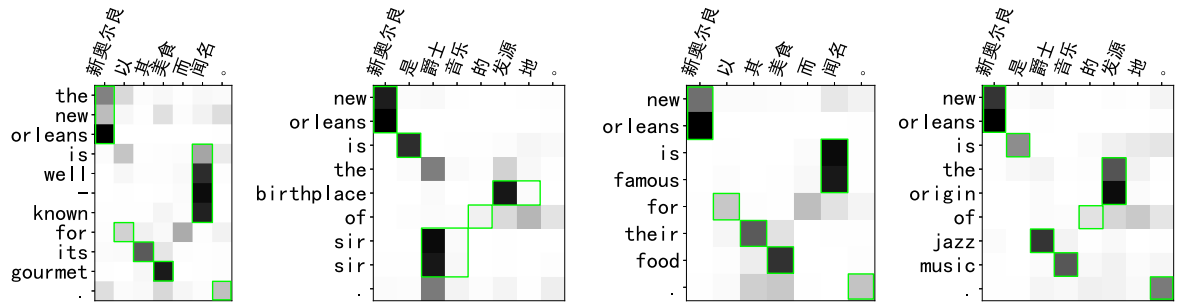
The results are shown in Figure 3. In the bins holding sentences no longer than 50, the BLEU scores of the two systems are close to each other. When the sentence length surpasses 50, RNMT shows its superiority over RNNsearch*. As the sentence length grows, the difference becomes increasingly large. This verifies the deduce that our method can not only memorize history information but capture the relationship between words, both of which are beneficial to translate long sentences.

5.5 Word Alignment

Systems	BLEU	AER
RNNsearch*	22.40	46.76
RNMT	**24.12**	**45.66**

Table 4: Comparison of alignment quality on NIST Zh-En translation task.

In this section, we will verify the translation performance of our model from another perspective. Intuitively, the better translation should have better alignment to the source sentence, so we evaluated the quality of the alignments derived from the attention module of the NMT using Alignment Error Rate (AER) (Och, 2003). We did this experiment on the artificially aligned dataset from Liu and Sun (2015) which contains 900 Zh-En sentence pairs. The alignments were got in this way for both RNNsearch*



(a) RNNsearch* (b) RNMT

Figure 4: Word alignment comparison. The green boxes show the manual golden alignments.

system and our system. When one target word was generated, we retained the alignment link with the highest probability α_{ij} in Equation 3.

The comparison results are shown in Table 4. It illustrates that our system RNMT can produce better translations than the baseline RNNsearch*, a difference of 1.72 BLEU points. Besides, the AER score is 1.1 points lower than the baseline model. Note that the smaller the AER score, the better the alignment quality.

Along with the translation results, we also produce the word alignment matrix based on each target word's attention probability distribution over the whole source sentence. Two source sentences are randomly sampled from websites, both comparisons between baseline alignment and improved alignment generated by RNNsearch* and RNMT are shown in Figure 4.

For the first example, from the view of source side, it is obviously unreasonable that the Chinese word *yi* is contributed to generate three discontinuous English words *the*, *is* and *for*, grammatical knowledge show that the word *yi* should be only aligned to the English word *for*, just like the result of our model. Besides, on the target's ground, if one Chinese word is translated into an English phrase, all words in the phrase should be aligned to the Chinese word, RNNsearch* model improperly aligns *new* and *is* to some other irrelevant words besides the correct one. When generating word *is*, almost the whole source sentence should be considered, our model gets more centralized alignment for it.

In the second case, unlike the baseline model, our model produces correct translation *jazz music* for *jueshi yinyue* and alignment. *the* together with *origin* is aligned to the source word *fayuan*, while RNNsearch* mistakenly aligns *the* to two source words almost with equal probability.

5.6 Translation Examples

As shown in Table 5, we give two example translations generated from baseline model and proposed model. Comparing the translation results between two systems, we can observe that RNNsearch* often miss some information of the source sentence, especially for the long sentence. Both of the sentences are complex sentences with long dependent adversative relation, for the first example, the baseline model forgets the information of the long distance clause about *women jinnian yizhi* $\cdots$ *toumingdu* and ignores to translate the second clause. It similarly happens that, when producing the target text for the second sample, RNNsearch* loses the information after *chengnuo dongaohui* and fails to capture the latter clause with adversative relation. In addition, another phenomenon observed is that the longer the source sentence is, it is easier to ignore important information for RNNsearch*. However, as can be seen from the boldfaced sections marked in results generated with RNMT, proposed model with CNN could captures more source information successfully.

Specifically, RNNs are skilled in modeling the order information of a sequence, while CNNs mainly focus on local features around some specific word. Both of them are weak to capture the long-distance dependency information, However, facts prove that proposed relation layer succeeds in alleviating the deficiencies of the two by integrating CNNs with bidirectional RNNs subtly.

Source	我们 近年 一直 倡导 “ 诚信 ” , 要 “ 打造 阳光 政府 ” , 要 尊重 公众 的 “ 知情权 ” , 要 提高 行政 “ 透明 度 ” , 然而 , 事实 距离 理想 还 有 很 大 差距 。
Reference	in recent years, we have been advocating ”integrity” and we want to ”forge a government-in-sunshine”, improve the ”transparency” of government administration, and respect the public's ”right to know”. however, the reality is still very far from ideal.
RNNsearch*	in recent years , we have always advocated ” honesty ” and ” build a sunshine government , ” and we must respect the public 's ” right to understand ” and to enhance the ” transparency ” of the ” transparency ” of the public .
RNMT	in recent years , we have advocated ” integrity ” and ” build up the sun . ” we should respect public ” right to know ” and improve the **” transparency ” of the public . however , there is still a big gap between reality and ideals .**
Source	经过 国际 奥委会 的 不懈 努力 , 意大利 方面 在 冬奥会 开幕 前 四 天 作出 让步 , 承诺 冬奥会 期间 警 方 不 会 进入 奥运村 搜查 运动员 驻地 , 但是 , 药检 呈 阳性 的 运动员 仍将 接受 意大利 检察 机关 的 调查 。
Reference	through the untiring efforts of the ioc, the italian side made concession four days before the winter olympics opened, promising that police would not enter the olympic village to raid athletes' quarters during the winter olympics, but athletes tested positive for drugs are still subject to investigations of italian prosecutors.
RNNsearch*	through the unremitting efforts of the ioc , the italian side made a concession four days prior to the opening of the international olympic committee .
RNMT	with the unremitting efforts of the international olympic committee , the italian side made a concession in four days before the opening of the $\langle$UNK$\rangle$ **and promised that the police would not be able to search for the athlete 's place during the opening period .**

Table 5: Translation examples.

6 Conclusion

As RNNs are not good at remembering the old history and cannot consider word relationship either, sometimes conventional NMT cannot get enough source information and hence emphasizes too much on the fluency of the target. As a result, it suffers from meaning-drift and generates “inaccurate” translation. Even so, NMT can still benefit from the recurrence of RNNs. In this paper, we propose to incorporate RNLs into the attentional NMT. The RNs employs CNNs to collect information around one word and explicitly connect each word with all the other words. In this way, it provides the opportunities for NMT to capture relationship between source words and hence leads to a better source representation. Our method can get better translation on the NIST Zh-En dataset and the WMT En-De dataset and can even outperform the system with supervised syntactic knowledge.

Acknowledgements

We highly appreciate the anonymous reviewers for their precious comments. This work was supported in part by National Natural Science Foundation of China (Nos. 61472428 and 61662077).

References

Roee Aharoni and Yoav Goldberg. 2017. Towards string-to-tree neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 132–140, Vancouver, Canada, July. Association for Computational Linguistics.

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

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967, Copenhagen, Denmark, September. Association for Computational Linguistics.

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 the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October. Association for Computational Linguistics.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 531–540, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. 2017a. A convolutional encoder model for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 123–135, Vancouver, Canada, July. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017b. Convolutional sequence to sequence learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252, International Convention Centre, Sydney, Australia, 06–11 Aug. PMLR.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger. 2017. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, July.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul. PMLR.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA, October. Association for Computational Linguistics.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.

Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. 2017. Modeling source syntax for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 688–697, Vancouver, Canada, July. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. 2017. A simple neural network module for relational reasoning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4974–4983. Curran Associates, Inc.

M. Schuster and K. K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, Nov.

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, Berlin, Germany, August. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August. Association for Computational Linguistics.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural mt learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas, November. Association for Computational Linguistics.

Jinsong Su, Shan Wu, Deyi Xiong, Yaojie Lu, Xianpei Han, and Biao Zhang. 2018. Variational recurrent neural machine translation. *arXiv preprint arXiv:1801.05119*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6000–6010. Curran Associates, Inc.

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

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July. Association for Computational Linguistics.

Yang Liu and Maosong Sun. 2015. Contrastive unsupervised word alignment with non-local features. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2295–2301. AAAI Press.