Recurrent Positional Embedding for Neural Machine Translation

Kehai Chen, Rui Wang, Masao Utiyama, and Eiichiro Sumita
National Institute of Information and Communications Technology (NICT), Kyoto, Japan
{khchen, wangrui, mutiyama, eiichiro.sumita}@nict.go.jp

Abstract

In the Transformer network architecture, positional embeddings are used to encode order dependencies into the input representation. However, this input representation only involves static order dependencies based on discrete numerical information, that is, are independent of word content. To address this issue, this work proposes a recurrent positional embedding approach based on word vector. In this approach, these recurrent positional embeddings are learned by a recurrent neural network, encoding word content-based order dependencies into the input representation. They are then integrated into the existing multi-head self-attention model as independent heads or part of each head. The experimental results revealed that the proposed approach improved translation performance over that of the state-of-the-art Transformer baseline in WMT'14 English-to-German and NIST Chinese-to-English translation tasks.

1 Introduction

Transformer translation systems (Vaswani et al., 2017), without recurrent and convolutional neural networks, rely on a positional embedding (**PE**) approach to encode order information into the input representation. PE is typically learned based on the position index of each word and is added to corresponding word embedding. This allows the Transformer to encode order dependencies between words in addition to the words themselves. Finally, the Transformer uses these combined vectors as the input to self-attention networks (**SANs**), achieving state-of-the-art translation performance with several language pairs (Vaswani et al., 2017; Dou et al., 2018; Zhang et al., 2018a; Marie et al., 2018, 2019).

In spite of their success, the input representation only involves static order dependencies based on discrete numerical information. That is, any word in the entire vocabulary has the same PE on the same position index. As a result, the dependencies encoded by the original PEs are independent of word content, which may further hinder the improvement of translation capacity. Recently, Chen et al. (2018) and Hao et al. (2019) introduced the additional source representation learned by an RNN-based encoder into Transformer to alleviate this issue, and reported improvements on the WMT'14 English-to-German translation task.

Inspired by their works (Chen et al., 2018; Hao et al., 2019), we propose an simple and efficient recurrent positional embedding approach to capture order dependencies based on word content in a sentence, thus learning a more effective sentence representation for the Transformer. In addition, we designed two simple multi-head self-attentions to introduce these learned RPEs and original input representation into the existing Transformer model for enhancing the sentence representation of Transformer. Experimental results on WMT'14 English-to-German and NIST Chinese-to-English translation tasks show that our models significantly improved translation performance over a strong Transformer baseline.

2 Background

2.1 Input Representation

In the Transformer network architecture (Vaswani et al., 2017), given a sentence of length J, the positional embedding of each word is firstly computed based on its position:

$$\begin{aligned} \mathbf{pe}_{(j,2i)} &= \sin(j/10000^{2i/d_{model}}), \\ \mathbf{pe}_{(j,2i+1)} &= \cos(j/10000^{2i/d_{model}}), \end{aligned} \quad (1)$$

* Corresponding author

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 1361–1367, Hong Kong, China, November 3–7, 2019.

where j is the word's numerical position index in the sentence and i is the dimension of position index. The word embedding $\mathbf{x}_j$ is then added with $\mathbf{pe}_j$ to get a combined embedding $\mathbf{v}_j$:

$$\mathbf{v}_j = \mathbf{x}_j + \mathbf{pe}_j. \quad (2)$$

As a result, there is a sequence of vectors $\{\mathbf{v}_1, \cdots, \mathbf{v}_J\}$ as the input to the encoder or decoder of the Transformer to learn the source or target sentence representations.

2.2 Self-Attention Mechanism

Following the input layer, the self-attention layer is used to learn the sentence representation for the Transformer (Vaswani et al., 2017). These combined vectors $\{\mathbf{v}_1, \cdots, \mathbf{v}_J\}$ are then packed into a query matrix $\boldsymbol{Q}$, their keys and values matrices $\boldsymbol{K}$ and $\boldsymbol{V}$. The output of the self-attention layer can be computed by Eq.(3):

$$\text{Att}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_{model}}})\boldsymbol{V}. \quad (3)$$

Moreover, self-attention can be further refined as multi-head self-attention to jointly attend to the information from different representation subspaces at different positions. Specifically, $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are linearly projected H times with different, learned linear projections to d_k and d_v dimensions, respectively. On each of these projected versions of queries, keys, and values, the attention function is performed in parallel, yielding d_v-dimensional output values. Take a single head as an example, the output of the h-th head $\boldsymbol{O}_h$ is computed by Eq.(4):

$$\boldsymbol{O}_h = \text{Att}(\boldsymbol{Q}\boldsymbol{W}_h^Q, \boldsymbol{K}\boldsymbol{W}_h^K, \boldsymbol{V}\boldsymbol{W}_h^V), \quad (4)$$

where the parameter matrices are $\boldsymbol{W}_h^Q \in \mathbb{R}^{d_{model} \times d_k}$, $\boldsymbol{W}_h^K \in \mathbb{R}^{d_{model} \times d_k}$, and $\boldsymbol{W}_h^V \in \mathbb{R}^{d_{model} \times d_v}$. For example, if there are H=8 heads and d_{model} is 512, d_k=d_v=512/8=64. Finally, the outputs of H heads are concatenated to serve as the sentence representation $\boldsymbol{S}$:

$$\boldsymbol{S} = \text{Concat}(\boldsymbol{O}_1, \cdots, \boldsymbol{O}_{H-1}, \boldsymbol{O}_H)\boldsymbol{W}^O, \quad (5)$$

where $\boldsymbol{W}^O \in \mathbb{R}^{Hd_v \times d_{model}}$ is a parameter matrix.

3 Recurrent Positional Embedding

We propose a recurrent positional embedding approach based on part of word embedding instead of numerical indices of words to capture order dependencies between words in a sentence. Specifically, the embedding of each word $\mathbf{x}_j$ is divided into two parts $\mathbf{x}_j^p$ and $\mathbf{x}_j^r$, and their dimensions are d_p and d_r (d_{model}=d_p+d_r), respectively. As a result, the sequence of word vectors $\{\mathbf{x}_1, \cdots, \mathbf{x}_J\}$ are spited into $\{\mathbf{x}_1^p, \cdots, \mathbf{x}_J^p\}$ and $\{\mathbf{x}_1^r, \cdots, \mathbf{x}_J^r\}$. An RNN with an non-linear projection layer is then designed to learn its recurrent state $\mathbf{r}_j$ for each word over $\{\mathbf{x}_1^r, \cdots, \mathbf{x}_J^r\}$:

$$\mathbf{r}_j = \text{Tanh}(f_{\text{RNN}}(\mathbf{x}_j^r, \mathbf{r}_{j-1})\mathbf{W}^r + \mathbf{b}^r), \quad (6)$$

where $\mathbf{W}^r \in \mathbb{R}^{d_r \times d_r}$ is a parameter matrix and $\mathbf{b}^r \in \mathbb{R}^{d_r}$ is a bias item.[1] Note that the $\mathbf{x}_j^r$ is derived from part of the word embedding $\mathbf{x}_j$. Finally, there is a sequence $\boldsymbol{R}$ =$\{\mathbf{r}_1, \cdots, \mathbf{r}_J\}$, called as recurrent positional embeddings (**RPEs**). In this work, a bidirectional RNN and a forward RNN (Bahdanau et al., 2015) are used to learn source RPEs and target RPEs, respectively. Noth that the RNN is also replaced by other neural networks for learning order dependency information, such as GRU (Cho et al., 2014), and SRU (Li et al., 2018a).

In addition, other sub-sequence $\{\mathbf{x}_1^p, \cdots, \mathbf{x}_J^p\}$ is used to gain the reduced dimension input representation $\boldsymbol{P}$=$\{\mathbf{p}_1, \cdots, \mathbf{p}_J\}$ according to the Section 2.1. Both of $\boldsymbol{R}$ and $\boldsymbol{P}$ will be together as the input to the encoder (or decoder) to learn a more effective source (or target) representation for the Transformer.

4 Neural Machine Translation with RPE

To make use of these learned RPEs, we propose two simple methods: RPE head (**RPEHead**) self-attention and mixed positional representation head (**MPRHead**) self-attention. Both of RPEHead and MPRHead can utilize RPEs to learn sentence representation for the Transformer.

4.1 RPEHead Self-Attention

For the RPEHead self-Attention, these learned RPEs are integrated into multi-head self-attention (Fig 1(a)) as several independent heads to learn the sentence representation, as shown in Fig 1(b).

First, we concatenate $\boldsymbol{P}$=$\{\mathbf{p}_1, \cdots, \mathbf{p}_J\}$ and $\boldsymbol{R}$ =$\{\mathbf{r}_1, \cdots, \mathbf{r}_J\}$ as a new combined input representation $\boldsymbol{\mathcal{T}}$. To perform the attention function in Eq.(3) over the combined input representation $\boldsymbol{\mathcal{T}}$, d_r and d_p are set to t*d_{model}/H

[1] The $\mathbf{r}_0$ is set as a zero vector.

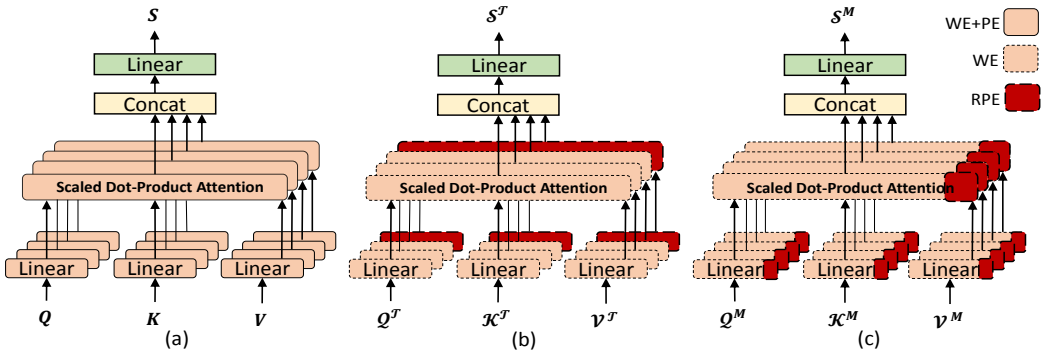

Figure 1: (a) Multi-Head Self-Attention; (b) RPEHead Self-Attention; (c) MPRHead Self-Attention.

and d_{model}-d_r in the process of learning RPEs, respectively. This guarantees that there are two types of heads: one type of head only contains RPE and other type of head only contains the original reduced dimension input representation. Second, the $\mathcal{T}$ is mapped to a new query matrix $\mathcal{Q}^{\mathcal{T}}$, and their keys and values matrices $\mathcal{K}^{\mathcal{T}}$ and $\mathcal{V}^{\mathcal{T}}$. According to Eq.(4), the output of each head is computed by Eq.(7):

$$\mathcal{O}_h^{\mathcal{T}} = \text{Att}(\mathcal{Q}^{\mathcal{T}}\boldsymbol{W}_h^{\mathcal{Q}}, \mathcal{K}^{\mathcal{T}}\boldsymbol{W}_h^{\mathcal{K}}, \mathcal{V}^{\mathcal{T}}\boldsymbol{W}_h^{\mathcal{V}}). \quad (7)$$

Therefore, the final sentence representation $\mathcal{S}^{\mathcal{T}}$ is formally represented as:

$$\mathcal{S}^{\mathcal{T}} = \text{Concat}(\mathcal{O}_1^{\mathcal{T}}, \cdots, \mathcal{O}_{H-1}^{\mathcal{T}}, \mathcal{O}_H^{\mathcal{T}})\boldsymbol{\mathcal{W}}^O. \quad (8)$$

4.2 MPRHead Self-Attention

Compared with the RPEHead model, the MPRHead model applies the multi-head mechanism to the RPEs. In other words, RPEs are encoded into the sentence representation from different vector sub-spaces, as shown in Fig 1(c).

To this end, each vector of $\boldsymbol{P}$ is divided into the H heads $\{\mathbf{p}_j^1, \cdots, \mathbf{p}_j^H\}$. Similarly, each vector of $\boldsymbol{R}$ is divided into the H heads $\{\mathbf{r}_j^1, \cdots, \mathbf{r}_j^H\}$. The corresponding heads for $\mathbf{p}_j$ and $\mathbf{r}_j$ are then concatenated as a combined sequence $\mathbf{pr}_j$, in turn:

$$\mathbf{pr}_j = \{\mathbf{p}_j^1 : \mathbf{r}_j^1, \cdots, \mathbf{p}_j^H : \mathbf{r}_j^H\}, \quad (9)$$

where ":" is the concatenation operation. All heads in $\mathbf{pr}_j$ are further concatenated as a mixed embedding $\mathbf{m}_j \in \mathbb{R}^{d_{model}}$ in turn. As a result, there is a new sequence of mixed embeddings:

$$\boldsymbol{M} = \{\mathbf{m}_1, \cdots, \mathbf{m}_J\}. \quad (10)$$

The $\boldsymbol{M}$ is mapped to a query matrix $\mathcal{Q}^{\boldsymbol{M}}$, and their keys and values matrices $\mathcal{K}^{\boldsymbol{M}}$ and $\mathcal{V}^{\boldsymbol{M}}$. According to Eq.(4) and Eq.(5), the final sentence representation $\mathcal{S}^{\boldsymbol{M}}$ is represented as:

$$\mathcal{O}_h^{\boldsymbol{M}} = \text{Att}(\mathcal{Q}^{\boldsymbol{M}}\boldsymbol{W}_h^{\mathcal{Q}}, \mathcal{K}^{\boldsymbol{M}}\boldsymbol{W}_h^{\mathcal{K}}, \mathcal{V}^{\boldsymbol{M}}\boldsymbol{W}_h^{\mathcal{V}}),$$
$$\mathcal{S}^{\boldsymbol{M}} = \text{Concat}(\mathcal{O}_1^{\boldsymbol{M}}, \cdots, \mathcal{O}_{H-1}^{\boldsymbol{M}}, \mathcal{O}_H^{\boldsymbol{M}})\boldsymbol{\mathcal{W}}^O. \quad (11)$$

Note that for both RPEHead and MPRHead models, the RPEs will be jointly learned with the existing Transformer architecture.

5 Experiments

5.1 Experimental Setup

The proposed methods were evaluated on the WMT'14 English to German (EN-DE) and NIST Chinese-to-English (ZH-EN) translation tasks. The ZH-EN training set includes *1.28* million bilingual sentence pairs from the LDC corpora, where the NIST06 and the NIST02/NIST03/NIST04 data sets were used as the development and test sets, respectively. The EN-DE training set includes *4.43* million bilingual sentence pairs of the WMT'14 corpora, where the newstest2013 and newstest2014 data sets were used as the development and test sets, respectively.

The BPE (Sennrich et al., 2016) was adopted and the vocabulary size was set as 32K. The dimension of all input and output layers was set to *512*, and that of the inner feedforward neural network layer was set to *2048*. The total heads of all multi-head modules were set to *8* in both encoder and decoder layers. In each training batch, there was a set of sentence pairs containing approximately *4096*4* source tokens and *4096*4* target tokens. For the other setting not mentioned, we followed the setting in Vaswani et al. (2017).

Baseline systems included a vanilla Transformer (Vaswani et al., 2017), Relative PEs (Shaw et al., 2018), and directional SAN (**DiSAN**) (Shen

System	Architecture	newstest2014	#Param
Existing NMT systems			
Vaswani et al. (2017)	Transformer (base)	27.3	65M
	Transformer (big)	28.4	213M
Chen et al. (2018)	RNMT+SAN	28.49	378.9M
Hao et al. (2019)	Transformer (base)+BiARN	28.21	97.4M
	Transformer (big)+BiARN	28.98	323.5M
Our NMT systems			
This work	Transformer (base)	27.25	97.35M
	+Relative PE	27.60	97.42M
	+DiSAN	27.66	97.39M
	+RPEHead	28.11*	97.84M
	+MPRHead	28.35*	97.72M
	Transformer (big)	28.22	272.6M
	+MPRHead	29.11*	289.1M

Table 1: Results for EN-DE translation task. The mark "*" after scores indicates that the model was significantly better than the baseline Transformer (base or big) at the significance level p <0.01 (Collins et al., 2005).

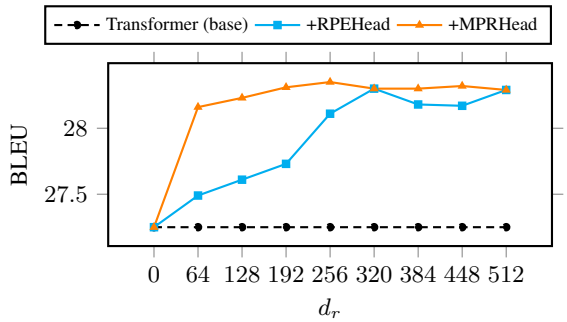

Figure 2: The BLEU scores on the different d_r.

et al., 2018). Besides, we reported results of the existing works (Vaswani et al., 2017; Chen et al., 2018; Shaw et al., 2018; Hao et al., 2019; Liu et al., 2019; Chen et al., 2019). We re-implemented the baseline Transformer, Relative PEs, and DiSAN models on the OpenNMT toolkit (Klein et al., 2017). All the models were trained for 200k batches and evaluated on a single V100 GPU. The multi-bleu.perl was used as the evaluation metric to obtain the case-sensitive 4-gram BLEU score of EN-DE and ZH-EN tasks.

5.2 Effect of RPEs

In this work, we extracted d_r dimensions of each word vector to learn recurrent embeddings. To explore the relation between d_r and translation performance, Figure 2 shows the translation performance on the different d_r. For +RPEHead (or +MPRHead), with the increasing in the dimension of d_r, BLEU scores are gradually increasing, but BLEU scores begin to decrease when d_r is more than 320 (or 256). In particular, +RPEHead and +MPRHead achieve the highest BLEU scores at d_r=320 and d_r=256, respectively. This means that the original partial input representation and our RPE can complement each other to improve translation performance.

5.3 Main Results

According to the results of Fig 2, d_r of RPEHead is set to 320 and d_r of MPRHead is set to 256. The main translation results are shown in Table 1.

1) For the proposed methods, both RPEHead (base) and MPRHead (base) outperformed Transformer (base), especially are better than +RPE and +DiSAN. This indicates that the learned RPEs are beneficial for the Transformer system.

2) Moreover, +MPRHead (base) performed better than RPEHead/RPE (base). The reason may be that adding RPEs into variant heads can encode order dependencies based word vectors from variant vector sub-spaces, which is one of the advantages of the multi-head mechanism. In particular, +MPRHead (base/big) is slightly better than Transformer (base/big)+BiARN. This denotes that the RPEs can more effective improve the performance of Transformer.

3) MPRHead (big) was superior to Transformer (big) significantly. MPRHead (base) achieved a comparable performance compared to Transformer (big) which contains approximately three times parameters, indicating that the proposed RPE is efficient.

In addition, Table 2 shows that the proposed models also gave similar improvements over the baseline system and the compared methods on the NIST ZH-EN task. These results indicate that our

System	Architecture	NIST02	NIST03	NIST04	#Param
Existing NMT systems					
Liu et al. (2019)	Share-private Embeddings	43.73	41.99	44.53	N/A
Chen et al. (2019)	Reordering Embeddings	47.54	46.56	47.27	78.3M
Our NMT systems					
This work	Transformer (base)	46.62	45.90	45.95	77.94M
	+Relative PEs (Shaw et al., 2018)	46.91	46.47	45.96	77.01M
	+RPEHead	47.06	46.21	46.32	78.04M
	+MPRHead	47.69*	47.27*	47.31*	78.04M
	Transformer (big)	47.88	47.58	47.37	243.7M
	+MPRHead	48.73*	48.61*	48.21*	245.5M

Table 2: Results for ZH-EN translation Task. The mark "*" after scores indicates that the model was significantly better than the baseline Transformer (base or big) at the significance level p <0.01 (Collins et al., 2005).

approach is a universal method for improving the translation of other language pairs.

5.4 Ablation Experiments

To further explore the effect of position information, PEs and RPEs were applied on the encoder (**Enc**) and decoder (**Dec**) side, respectively. Table 3 shows the results of ablation experiments on the EN-DE newstest2014 test set:

	newstest2014	#Speed
Transformer (base)	27.25	11.7K
- PE of Dec	26.73	11.7K
- PE of Enc	10.71	11.7K
RPEHead (base)	28.11	10.5K
- RPE&PE of Dec	27.54	11.2K
- RPE&PE of Enc	11.16	10.7K
MPRHead (base)	28.35	9.9K
- RPE&PE of Dec	27.74	11.2K
- RPE/PE of Enc	10.89	10.7K
- RPE&PE of Enc&Dec	10.43	11.7K

Table 3: Ablation experiments of position information. "#Speed" denotes the training speed (tokens/second).

1) For Enc/Dec/Enc&Dec, the proposed RPEHead/MPRHead outperformed the Transformer in corresponding ablation settings, indicating that our methods worked well on both the encoder and decoder.

2) For PE/RPEs, performance became slightly lower when PE was removed from the decoder. However, when PE was removed from the encoder, the translation performance drastically decreased. We think that the source sentence representation is more sensitive to position information than the target sentence representation. It is possible that each hidden state in the decoder takes the previous hidden state into consideration, which would be somewhat similar to the proposed RPEs. In contrast, the encoder would not be expected to contain this mechanism. Therefore, the position information would be more important in the encoder than the decoder.

3) The training speeds of the proposed RPEHead (base) and MPRHead (base) were just slightly slower than those of the vanilla Transformer (base), because we nearly did not introduce additional (less than 2%) parameters.

6 Conclusion and Future Work

In this paper, we presented a recurrent embedding to capture the order dependencies in a sentence. Empirical results show that this method can improve the performance of NMT efficiently. In future work, we plan to extend this method to unsupervised NMT (Sun et al., 2019) and other natural language processing tasks, such as dependency parsing (Li et al., 2018b) and reading comprehension (Zhang et al., 2018b).

Acknowledgments

We are grateful to the anonymous reviewers and the area chair for their insightful comments and suggestions. This work was partially conducted under the program "Research and Development of Enhanced Multilingual and Multipurpose Speech Translation Systems" of the Ministry of Internal Affairs and Communications (MIC), Japan. Masao Utiyama is partly supported by JSPS KAKENHI Grant Number 19H05660. Rui Wang was partially supported by JSPS grant-in-aid for early-career scientists (19K20354): "Unsupervised Neural Machine Translation in Universal Scenarios" and NICT tenure-track researcher startup fund "Toward Intelligent Machine Translation".

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA.

Kehai Chen, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2019. Neural machine translation with reordering embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1787–1799, Florence, Italy. Association for Computational Linguistics.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86, Melbourne, Australia. 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. 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. Association for Computational Linguistics.

Zi-Yi Dou, Zhaopeng Tu, Xing Wang, Shuming Shi, and Tong Zhang. 2018. Exploiting deep representations for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4253–4262, Brussels, Belgium. Association for Computational Linguistics.

Jie Hao, Xing Wang, Baosong Yang, Longyue Wang, Jinfeng Zhang, and Zhaopeng Tu. 2019. Modeling recurrence for transformer. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1198–1207, Minneapolis, Minnesota. Association for Computational Linguistics.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Jian Li, Zhaopeng Tu, Baosong Yang, Michael R. Lyu, and Tong Zhang. 2018a. Multi-head attention with disagreement regularization. In *Proceedings of the 2018 Conferenctemporale on Empirical Methods in Natural Language Processing*, pages 2897–2903, Brussels, Belgium. Association for Computational Linguistics.

Zuchao Li, Jiaxun Cai, Shexia He, and Hai Zhao. 2018b. Seq2seq dependency parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3203–3214, Santa Fe, New Mexico, USA.

Xuebo Liu, Derek F. Wong, Yang Liu, Lidia S. Chao, Tong Xiao, and Jingbo Zhu. 2019. Shared-private bilingual word embeddings for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3613–3622, Florence, Italy. Association for Computational Linguistics.

Benjamin Marie, Haipeng Sun, Rui Wang, Kehai Chen, Atsushi Fujita, Masao Utiyama, and Eiichiro Sumita. 2019. NICT's unsupervised neural and statistical machine translation systems for the WMT19 news translation task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 294–301, Florence, Italy. Association for Computational Linguistics.

Benjamin Marie, Rui Wang, Atsushi Fujita, Masao Utiyama, and Eiichiro Sumita. 2018. NICT's neural and statistical machine translation systems for the WMT18 news translation task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 449–455, Belgium, Brussels. 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. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. Disan:

Directional self-attention network for rnn/cnn-free language understanding. In *AAAI Conference on Artificial Intelligence*.

Haipeng Sun, Rui Wang, Kehai Chen, Masao Utiyama, Eiichiro Sumita, and Tiejun Zhao. 2019. Unsupervised bilingual word embedding agreement for unsupervised neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1235–1245, Florence, Italy. Association for Computational Linguistics.

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 5998–6008. Curran Associates, Inc.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2018a. Accelerating neural transformer via an average attention network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Melbourne, Australia. Association for Computational Linguistics.

Zhuosheng Zhang, Yafang Huang, and Hai Zhao. 2018b. Subword-augmented embedding for cloze reading comprehension. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1802–1814, Santa Fe, New Mexico, USA. Association for Computational Linguistics.