Lattice-Based Transformer Encoder for Neural Machine Translation

Fengshun Xiao[1,2], **Jiangtong Li**[2,3], **Hai Zhao**[1,2,*] **Rui Wang**[4], **Kehai Chen**[4]

[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Key Laboratory of Shanghai Education Commission for Intelligent Interaction
and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, China
[3]College of Zhiyuan, Shanghai Jiao Tong University, China
[4]National Institute of Information and Communications Technology (NICT)
{felixxiao, keep_moving-lee}@sjtu.edu.cn,
zhaohai@cs.sjtu.edu.cn, {wangrui, khchen}@nict.go.jp

Abstract

Neural machine translation (NMT) takes deterministic sequences for source representations. However, either word-level or subword-level segmentations have multiple choices to split a source sequence with different word segmentors or different subword vocabulary sizes. We hypothesize that the diversity in segmentations may affect the NMT performance. To integrate different segmentations with the state-of-the-art NMT model, Transformer, we propose lattice-based encoders to explore effective word or subword representation in an automatic way during training. We propose two methods: 1) lattice positional encoding and 2) lattice-aware self-attention. These two methods can be used together and show complementary to each other to further improve translation performance. Experiment results show superiorities of lattice-based encoders in word-level and subword-level representations over conventional Transformer encoder.

1 Introduction

Neural machine translation (NMT) has achieved great progress with the evolution of model structures under an encoder-decoder framework (Sutskever et al., 2014; Bahdanau et al., 2014). Recently, the self-attention based Transformer model has achieved state-of-the-art performance on multiple language pairs (Vaswani et al., 2017; Marie et al., 2018). Both representations of source and target sentences in
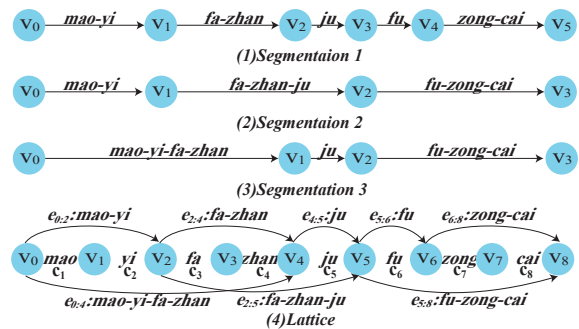

Figure 1: Incorporating three different segmentation for a lattice graph. The original sentence is "*mao-yi-fa-zhan-ju-fu-zong-cai*". In Chinese it is "贸易发展局副总裁". In English it means "*The vice president of Trade Development Council*"

NMT can be factorized in character (Costa-Jussa and Fonollosa, 2016), word (Sutskever et al., 2014), or subword (Sennrich et al., 2015) level. However, only using 1-best segmentation as inputs limits NMT encoders to express source sequences sufficiently and reliably. Many East Asian languages, including Chinese are written without explicit word boundary, so that their sentences need to be segmented into words firstly (Zhao et al., 2019; Cai et al., 2017; Cai and Zhao, 2016; Zhao et al., 2013; Zhao and Kit, 2011). By different segmentors, each sentence can be segmented into multiple forms as shown in Figure 1. Even for those alphabetical languages with clear word boundary like English, there is still an issue about selecting a proper subword vocabulary size, which determines the segmentation granularities for word representation.

In order to handle this problem, Morishita et al. (2018) used hierarchical subword features to represent sequence with different subword granularities. Su et al. (2017) proposed the first word-lattice based recurrent neural network

(RNN) encoders which extended Gated Recurrent Units (GRUs) (Cho et al., 2014) to take in multiple sequence segmentation representations. Sperber et al. (2017) incorporated posterior scores to Tree-LSTM for building a lattice encoder in speech translation. All these existing methods serve for RNN-based NMT model, where lattices can be formulized as directed graphs and the inherent directed structure of RNN facilitates the construction of lattice. Meanwhile, the self-attention mechanism is good at learning the dependency between characters in parallel, which can partially compare and learn information from multiple segmentations (Cherry et al., 2018). Therefore, it is challenging to directly apply the lattice structure to Transformer.

In this work, we explore an efficient way of integrating lattice into Transformer. Our method can not only process multiple sequences segmented in different ways to improve translation quality, but also maintain the characteristics of parallel computation in the Transformer.

2 Background

2.1 Transformer

Transformer stacks self-attention and point-wise, fully connected layers for both encoders and decoders. Decoder layers also have another sub-layer which performs attention over the output of the encoder. Residual connections around each layer are employed followed by layer normalization (Ba et al., 2016).

To make use of the order of the sequence, Vaswani et al. (2017) proposed Positional Encodings to indicate the absolute or relative position of tokens in input sequence which are calculated as:
$$p_{(j,2i)} = sin(j/10000^{2i/d})$$
$$p_{(j,2i+1)} = cos(j/10000^{2i/d}),$$
where j is the position, i is the dimension and d is the model dimension. Then positional encodings $p_{1:M} = \{p_1, ..., p_M\}$ are added to the embedding of each token $t_{1:M} = \{t_1, ..., t_M\}$ and are propagated to higher layers via residual connections.

2.2 Self-Attention

Transformer employs H attention heads to perform self-attention over a sequence individually and finally applies concatenation and linear transformation to the results from

	Conditions	Explanation
lad	$i < j = p < q$	$e_{i:j}$ is left adjacent to $e_{p:q}$.
rad	$p < q = i < j$	$e_{i:j}$ is right adjacent to $e_{p:q}$.
inc	$i \leq p < q \leq j$	$e_{i:j}$ includes $e_{p:q}$.
ind	$p \leq i < j \leq q$	$e_{i:j}$ is included in $e_{p:q}$.
its	$i < p < j < q$ or $p < i < q < j$	$e_{i:j}$ is intersected with $e_{p:q}$.
pre	$i < j < p < q$	$e_{i:j}$ is preceding edge to $e_{p:q}$.
suc	$p < q < i < j$	$e_{i:j}$ is succeeding edge to $e_{p:q}$.

Table 1: Relations possibly satisfied by any two different edges $e_{i:j}$ and $e_{p:q}$ in the lattice. Note that two equal signs cannot stand at the same time in condition inequality for **inc** and **ind**.

.

each head, which is called multi-head attention (Vaswani et al., 2017). Every single head attention in multi-head attention is calculated in a scaled dot product form:

$$u_{ij} = \frac{(t_i W^Q)(t_j W^K)^T}{\sqrt{d}}, \qquad (1)$$

where d is the model dimension, $t_{1:M}$ is the input sequence and u_{ij} are normalized by a softmax function:

$$\alpha_{ij} = \frac{\exp(u_{ij})}{\sum_{k=1}^{M} \exp(u_{ik})}, \qquad (2)$$

and α_{ij} are used to calculate the final output hidden representations:

$$o_i = \sum_{j=1}^{M} \alpha_{ij}(t_j W^V), \qquad (3)$$

where $o_{1:M}$ is outputs and W^Q, W^K, and W^V are learnable projections matrices for query, key, and value in a single head, respectively.

3 Models

3.1 Lattices

Lattices can represent multiple segmentation sequences in a directed graph, as they merge the same subsequence of all candidate subsequences using a compact way.

As shown in Figure 1, we follow Su et al. (2017) to apply different segmentator to segment an element[1] sequence $c_{1:N} = \{c_1, c_2, ..., c_N\}$ into different word or subword sequences to construct a lattice $G = \langle V, E \rangle$, a directed, connected, and acyclic graph, where V is node set and E is edge

---

[1] Character for word lattice and minimum subword unit in our predefined subword segmentations for subword lattice.
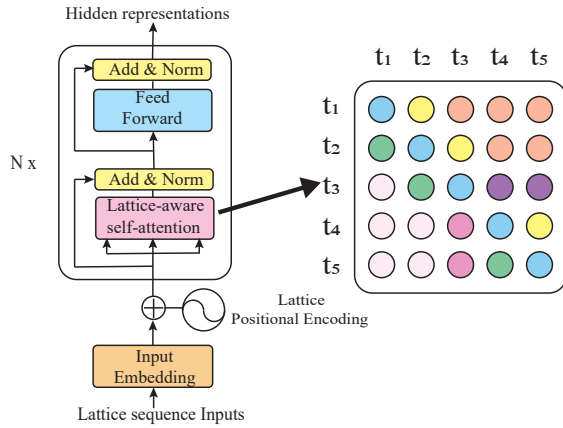
Figure 2: The architecture of lattice-based Transformer encoder. Lattice positional encoding is added to the embeddings of lattice sequence inputs. Different colors in lattice-aware self-attention indicate different relation embeddings.

set, node $v_i \in V$ denotes the gap between c_i and c_{i+1}, edge $e_{i:j} \in E$ departing from v_i and arrives at v_j ($i < j$) indicates a possible word or subword unit covering subsequence $c_{i+1:j}$.

All the edges in the lattice G are the actual input tokens for NMT. For two different edges $e_{i:j}$ and $e_{p:q}$, all possible relations can be enumerated as in Table 1.

3.2 Lattice-Based Encoders

We place all edges E in the lattice graph into an input sequence $t_{1:M} = \{t_1, t_2, ..., t_M\}$ for Transformer; then we modify the positional encoding to indicate the positional information of input tokens, namely all edges in the lattice graph. In addition, we propose a lattice-aware self-attention to directly represent position relationship among tokens. The overall architecture is shown in Figure 2.

Lattice Positional Encoding (LPE) Original positional encoding indicates the order of the sequence in an ascending form $\{p_1, p_2, ..., p_M\}$. We hypothesize that increasing positional encodings can indicate the order of sequential sentence. As shown in Figure 3, we scan a source sequence by element $c_{1:N} = \{c_1, c_2, ..., c_N\}$ (for example, c_i is character in Figure 3) and record their position $p_{1:N} = \{p_1, p_2, ..., p_N\}$. Then we use the positional encoding of the first element in lattice edge to represent current token's position, which can ensure that every edge in each path departing from v_0 and arriving at v_N in lattice will
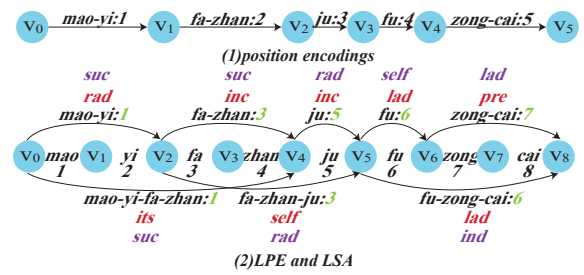

Figure 3: Lattice positional encoding p_{i+1} (in green) for edge $e_{i:j}$ in the lattice graph and the relation embeddings r in lattice-aware self-attention based on the timestep of token *fa-zhan-ju* (in red) and *fu* (in purple).

have an increasing positional encoding order. The property mentioned above is easy to prove, since start and end points v_i, v_j of each edge $e_{i:j}$ strictly satisfy $i < j$ and next edge $e_{j:k}$ will start from v_j and thus get a larger positional encoding.

Formally, for any input token t_k, namely edge $e_{i:j}$ covering elements $c_{i+1:j}$, positional encoding p_{i+1} will be used to represent its position and be added to its embedding.

Lattice-aware Self-Attention (LSA) We also directly modify self-attention to a lattice-aware way which makes self-attention aware of the relations between any two different edges. We modified Equations (1) and (3) in the same way of Shaw et al. (2018) to indicate edge relation:

$$e_{ij} = \frac{(t_i W^Q)(t_j W^K + r_{ij}^K)^T}{\sqrt{d}}, \quad (4)$$

$$o_i = \sum_{j=1}^{M} \alpha_{ij}(t_j W^V + r_{ij}^V), \quad (5)$$

where r_{ij}^K and r_{ij}^V are relation embeddings which are added to the keys and values to indicate relation between input tokens t_i and t_j, namely edges $e_{p:q}$ and $e_{k:l}$ in lattice graph, respectively.

To facilitate parallel computation, we add an additional embedding (**self**) for a token when it is conducted dot-product attention with itself, so we train eight (seven in Table 1) different relation embeddings $a_{1:8}^V$ and $a_{1:8}^K$ as look-up table for keys and values, respectively. r_{ij}^K and r_{ij}^V can look up for $a_{1:8}^V$ and $a_{1:8}^K$ based on the relation between t_i and t_j. Figure 3 shows an example of embeddings in lattice-aware self-attentions based on the timestep of token *fa-zhan-ju* and *fu*.

System	Input	MT05	MT02	MT03	MT04	MT06	MT08	ALL
	PKU	31.42	34.68	33.08	35.32	31.61	23.58	31.76
RNN	CTB	31.38	34.95	32.85	35.44	31.75	23.33	31.78
	MSR	29.92	34.49	32.06	35.10	31.23	23.12	31.35
Lattice-RNN	Lattice	32.40	35.75	34.32	36.50	32.77	24.84	32.95
	PKU	41.67	43.61	41.62	43.66	40.25	31.62	40.24
Transformer	CTB	41.87	43.72	42.11	43.58	40.41	31.76	40.35
	MSR	41.17	43.11	41.38	43.60	39.67	31.02	39.87
Transformer + LPE		42.37	43.71	42.67	44.43	41.14	32.09	40.93$^{\uparrow}$
Transformer + LSA	Lattice	42.28	43.56	**42.73**	43.81	41.01	32.39	40.77$^{\uparrow}$
Transformer + LPE + LSA		**42.65**	**44.14**	42.24	**44.81**	**41.37**	**32.98**	**41.26**$^{\uparrow}$

Table 2: Evaluation of translation performance on NIST Zh-En dataset. RNN and Lattice-RNN results are from (Su et al., 2017). We highlight the highest BLEU score in bold for each set. $\uparrow$ indicates statistically significant difference ($p <0.01$) from best baseline.

Since self-attention is computed parallelly, we generate a matrix with all lattice embeddings in it for each sentence which can be easily incorporated into standard self-attention by matrix multiplication. We use different relation embeddings for different Transformer layers but share the same one between different heads in a single layer.

4 Experiments

4.1 Setup

We conducted experiments on the NIST Chinese-English (Zh-En) and IWSLT 2016 English-German (En-De) datasets. The Zh-En corpus consists of 1.25M sentence pairs and the En-De corpus consists of 191K sentence pairs. For Zh-En task, we chose the NIST 2005 dataset as the validation set and the NIST 2002, 2003, 2004, 2006, and 2008 datasets as test sets. For En-De task, tst2012 was used as validation set and tst2013 and tst2014 were used as test sets. For both tasks, sentence pairs with either side longer than 50 were dropped. We used the case-sensitive 4-gram NIST BLEU score (Papineni et al., 2002) as the evaluation metric and sign-test (Collins et al., 2005) for statistical significance test.

For Zh-En task, we followed Su et al. (2017) to use the toolkit[2] to train segmenters on PKU, MSR (Emerson, 2005), and CTB corpora (Xue et al., 2005), then we generated word lattices with different segmented training data. Both source and target vocabularies are limited to 30K. For En-De task, we adopted 8K, 16K and 32K

BPE merge operations (Sennrich et al., 2015) to get different segmented sentences for building subword lattices. 16K BPE merge operations are employed on the target side.

We set batch size to 1024 tokens and accumulated gradient 16 times before a back-propagation. During training, we set all dropout to 0.3 and chose the Adam optimizer (Kingma and Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$ for parameters tuning. During decoding, we used beam search algorithm and set the beam size to 20. All other configurations were the same with Vaswani et al. (2017). We implemented our model based on the OpenNMT (Klein et al., 2017) and trained and evaluated all models on a single NVIDIA GeForce GTX 1080 Ti GPU.

4.2 Overall Performance

From Table 2, we see that our LPE and LSA models both outperform the Transformer baseline model of 0.58 and 0.42 BLEU respectively. When we combine LPE and LSA together, we get a gain of 0.91 BLEU points. Table 3 shows that our method also works well on the subword level.

The base Transformer system has about 90M parameters and our LPE and LSA models introduce 0 and 6k parameters over it, respectively, which shows that our lattice approach improves Transformer with little parameter accumulation.

During training, base Transformer performs about 0.714 steps per second while LPE + LSA model can process around 0.328. As lattice-based method usually seriously slows down the training, our lattice design and implementation over the Transformer only shows moderate efficiency

---

[2]https://nlp.stanford.edu/software/segmenter.html#Download

System	Input	tst2012	tst2013	tst2014
RNN	16k	26.24	28.22	24.17
Transformer	8k	27.31	29.56	25.57
	16k	27.35	29.02	25.12
	32k	27.15	28.61	24.88
+ LPE		27.34	29.48	25.88[↑]
+ LSA	Lattice	27.44	29.73[↑]	25.65
+ LPE + LSA		**27.76**	**30.28**[↑]	**26.22**[↑]

Table 3: Evaluation of translation performance on IWSLT2016 En-De dataset. RNN results are reported from Morishita et al. (2018). ↑ indicates statistically significant difference ($p <0.01$) from best baseline.
.

Systems	PE	PE + LSA
ALL	40.54	40.90

Table 4: Translation performance (BELU score) with normal positional encodings and normal positional encodings with LSA model on NIST Zh-En dataset.

reduction.

4.3 Analysis[3]

Effect of Lattice-Based Encoders To show the effectiveness of our method, we placed all edges in the lattice of a single sequence in a relative right order based on their first character, then we applied normal positional encodings (PE) to the lattice inputs on our base Transformer model. As shown in Table 4, our LPE and LSA method outperforms normal positional encodings by 0.39 and 0.23 BLEU respectively which shows that our methods are effective.

Complementary of LPE and LSA Our LPE method allows edges in all paths in an increasing positional encoding order which seems to focus on long-range order but ignore local disorder. While our LSA method treats all preceding and succeeding edges equally which seems to address local disorder better but ignore long-range order. To show the complementary of these two methods, we also placed all edges of lattice in a single sequence in a relative right order based on their first character and use normal positional encodings and our LSA method; we obtained a BLEU of 40.90 which is 0.13 higher than single LSA model. From this, we can see that long-range position information is indeed beneficial to our LSA model.

---

[3]All analysis experiments conducted on NIST dataset.

5 Related Work

Neural network based methods have been applied to several natural language processing tasks (Li et al., 2018; Zhang et al., 2019; Chen et al., 2018, 2017; Li et al., 2019; He et al., 2018; Zhou and Zhao, 2019), especially to NMT (Bahdanau et al., 2015; Wang et al., 2017a,b, 2018; Wang et al., 2018; Zhang et al., 2018; Zhang and Zhao, 2019).

Our work is related to the source side representations for NMT. Generally, the NMT model uses the word as a basic unit for source sentences modeling. In order to obtain better source side representations and avoid OOV problems, recent research has modeled source sentences at character level (Ling et al., 2015; Costa-Jussa and Fonollosa, 2016; Yang et al., 2016; Lee et al., 2016), subword level (Sennrich et al., 2015; Kudo, 2018; Wu and Zhao, 2018) and mixed character-word level (Luong and Manning, 2016). All these methods show better translation performance than the word level model.

As models mentioned above only use 1-best segmentation as inputs, lattice which can pack many different segmentations in a compact form has been widely used in statistical machine translation (SMT) (Xu et al., 2005; Dyer et al., 2008) and RNN-based NMT (Su et al., 2017; Sperber et al., 2017). To enhance the representaions of the input, lattice has also been applied in many other NLP tasks such as named entity recognition (Zhang and Yang, 2018), Chinese word segmentation (Yang et al., 2019) and part-of-speech tagging (Jiang et al., 2008; Wang et al., 2013).

6 Conclusions

In this paper, we have proposed two methods to incorporate lattice representations into Transformer. Experimental results in two datasets on word-level and subword-level respectively validate the effectiveness of the proposed approaches.

Different from Veličković et al. (2017), our work also provides an attempt to encode a simple labeled graph into Transformer and can be used in any tasks which need Transformer encoder to learn sequence representation.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

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 (ICLR 2015)*.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for Chinese. *arXiv preprint arXiv:1606.04300*.

Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for Chinese. *arXiv preprint arXiv:1704.07047*.

Kehai Chen, Rui Wang, Masao Utiyama, Lemao Liu, Akihiro Tamura, Eiichiro Sumita, and Tiejun Zhao. 2017. Neural machine translation with source dependency representation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 2846–2852.

Kehai Chen, Rui Wang, Masao Utiyama, Eiichiro Sumita, and Tiejun Zhao. 2018. Syntax-directed attention for neural machine translation. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI 2018)*, pages 4792–4799.

Colin Cherry, George Foster, Ankur Bapna, Orhan Firat, and Wolfgang Macherey. 2018. Revisiting character-based neural machine translation with capacity and compression. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 4295–4305.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

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

Marta R Costa-Jussa and José AR Fonollosa. 2016. Character-based neural machine translation. *arXiv preprint arXiv:1603.00810*.

Christopher Dyer, Smaranda Muresan, and Philip Resnik. 2008. Generalizing word lattice translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL 2008)*, pages 1012–1020.

Thomas Emerson. 2005. The second international Chinese word segmentation bakeoff. In *Proceedings of the fourth SIGHAN workshop on Chinese Language Processing*, pages 123–133.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 2061–2071.

Wenbin Jiang, Haitao Mi, and Qun Liu. 2008. Word lattice reranking for Chinese word segmentation and part-of-speech tagging. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 385–392.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 66–75.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2016. Fully character-level neural machine translation without explicit segmentation. *arXiv preprint arXiv:1610.03017*.

Zuchao Li, Jiaxun Cai, Shexia He, and Hai Zhao. 2018. Seq2seq dependency parsing. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 3203–3214.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*.

Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W. Black. 2015. Character-based neural machine translation. *arXiv preprint arXiv:1511.04586*.

Minh-Thang Luong and Christopher D. Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1054–1063.

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, Volume 2: Shared Task Papers*, pages 453–459.

Makoto Morishita, Jun Suzuki, and Masaaki Nagata. 2018. Improving neural machine translation by incorporating hierarchical subword features. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 618–629.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2017. Neural lattice-to-sequence models for uncertain inputs. *arXiv preprint arXiv:1704.00559*.

Jinsong Su, Zhixing Tan, Deyi Xiong, Rongrong Ji, Xiaodong Shi, and Yang Liu. 2017. Lattice-based recurrent neural network encoders for neural machine translation. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI 2017)*, pages 3302–3308.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 28th Conference on Neural Information Processing Systems (NIPS 2014)*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31stst Conference on Neural Information Processing Systems (NIPS 2017)*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Rui Wang, Andrew Finch, Masao Utiyama, and Eiichiro Sumita. 2017a. Sentence embedding for neural machine translation domain adaptation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 560–566.

Rui Wang, Masao Utiyama, Andrew Finch, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2018. Sentence selection and weighting for neural machine translation domain adaptation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26:1727–1741.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017b. Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 1482–1488.

Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2018. Dynamic sentence sampling for efficient training of neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 298–304.

Zhiguo Wang, Chengqing Zong, and Nianwen Xue. 2013. A lattice-based framework for joint Chinese word segmentation, POS tagging and parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 623–627.

Yingting Wu and Hai Zhao. 2018. Finding better subword segmentation for neural machine translation. In *The Seventeenth China National Conference on Computational Linguistics (CCL 2018)*, pages 53–64.

Jia Xu, Evgeny Matusov, Richard Zens, and Hermann Ney. 2005. Integrated Chinese word segmentation in statistical machine translation. In *International Workshop on Spoken Language Translation (IWSLT 2005)*.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The Penn Chinese TreeBank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(2):207–238.

Jie Yang, Yue Zhang, and Shuailong Liang. 2019. Subword encoding in lattice LSTM for Chinese word segmentation. In *Proceedings of the 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2019)*.

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2016. A character-aware encoder for neural machine translation. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016)*, pages 3063–3070.

Huan Zhang and Hai Zhao. 2019. Minimum divergence vs. maximum margin: An empirical comparison on seq2seq models. In *Proceedings of the Seventh International Conference on Learning Representations (ICLR 2019)*.

Yue Zhang and Jie Yang. 2018. Chinese NER using lattice LSTM. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 1554–1564.

Zhisong Zhang, Rui Wang, Masao Utiyama, Eiichiro Sumita, and Hai Zhao. 2018. Exploring recombination for efficient decoding of neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 4785–4790.

Zhuosheng Zhang, Yafang Huang, and Hai Zhao. 2019. Neural-based pinyin-to-character conversion with adaptive vocabulary. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*.

Hai Zhao, Deng Cai, Changning Huang, and Chunyu Kit. 2019. Chinese word segmentation: Another decade review (2007-2017). *arXiv preprint arXiv:1901.06079*.

Hai Zhao and Chunyu Kit. 2011. Integrating unsupervised and supervised word segmentation: The role of goodness measures. *Information Sciences*, 181(1):163–183.

Hai Zhao, Masao Utiyama, Eiichiro Sumita, and Bao-Liang Lu. 2013. An empirical study on word segmentation for Chinese machine translation. In *International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2013)*, pages 248–263.

Junru Zhou and Hai Zhao. 2019. Head-driven phrase structure grammar parsing on Penn Treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*.