Kingsoft's Neural Machine Translation System for WMT19

Xinze Guo, Chang Liu, Xiaolong Li, Yiran Wang
Guoliang Li, Feng Wang, Zhitao Xu, Liuyi Yang, Li Ma
Changliang Li*
Kingsoft AI Lab
{guoxinze,liuchang10,lixiaolong2,wangyiran3,liguoliang,
wangfeng5,xuzhitao,yangliuyi,mali5,lichangliang}@kingsoft.com

Abstract

This paper describes the Kingsoft AI Lab's submission to the WMT2019 news translation shared task. We participated in two language directions: English→Chinese and Chinese→English. For both language directions, we trained several variants of Transformer models using the provided parallel data enlarged with a large quantity of back-translated monolingual data. The best translation result was obtained with ensemble and reranking techniques. According to automatic metrics (BLEU) our Chinese→English system reached the second highest score, and our English→Chinese system reached the second highest score for this subtask.

1 Introduction

In recent years, the development of sequence-to-sequence (seq2seq) models have changed the field of machine translation a lot. This kind of models replaced traditional statistical approaches with neural machine translation (NMT) systems which is based on the encoder-decoder framework. Two years ago, the Transformer model, which is based on the multi-head attention mechanism and feedforward networks, has further advanced the field of NMT by improving the translation quality and speed of convergence (Vaswani et al., 2017; Ahmed et al., 2017). Until now, a variety of NMT models and advanced techniques have been proposed, leading to better performance of machine translation. We participated in the WMT19 shared task: the machine translation of news on English↔Chinese language pairs. This paper describes the NMT systems we submitted for the WMT19 Chinese→English and English→Chinese machine translation tasks. For data augmentation, we selected a subset of monolingual corpus as additional datasets and applied back translation to augment our training corpus. The baseline model in our system was based on the Transformer architecture. In order to improve the single system's performance, we experimented with some research findings such as Transformer with Relative Position Attention (Shaw et al., 2018) and Dynamic Convolution Networks (Wu et al., 2019).

We also proposed our own model architectures and applied them in the tasks. These architectures improve translation quality a lot and will be described in the next section. For further improvement, we tried different multi-system based techniques, such as model ensembling and model reranking. These techniques can improve translation performance on the basis of a very strong single system. At the same time, we also designed some specific strategies to deal with problems during ensembling, such as the overflow of memory space and the slow decoding speed. As a result, our Chinese→English system achieved the second highest cased BLEU score among all 15 submitted constrained systems, and our English→Chinese system ranked the second out of 12 submitted systems.

2 Model Features

This section describes five different model architectures applied to translation tasks. Two of them come from public research works, while the other three come from our works. The Transformer was used as our baseline system.

2.1 Transformer with Relative Position

We used relative position representation in self-attention mechanism (Shaw et al., 2018) of both the encoder side and decoder side. Originally, the Transformer only uses the absolute position information that calculated by sinusoidal functions,

*Corresponding author

Proceedings of the Fourth Conference on Machine Translation (WMT), Volume 2: Shared Task Papers (Day 1) pages 196–202

lacking of considering the relative position representation efficiently. Thus, it is an alternative approach to incorporate relative position representation in self-attention mechanism. In contrast to the absolute position, the relative position representation is invariant to the sentence length. We compared the translation results between whether using this feature or not, and found that model with relative position representation performs better. We conducted an implement in Fairseq[1] as an additional architecture with precise tuning. Experiments showed that this architecture leads to faster convergence and better performance.

2.2 Dynamic Convolution Network

Different from Transformer based on self-attention mechanism, Dynamic Convolution Network (Wu et al., 2019) uses a convolution network to replace the self-attention mechanism in the model framework. It predicts separate convolution kernels based solely on the current time-step in order to determine the importance of context elements. In other word, a Dynamic Convolution Network has kernels that vary over time as a learned function of the individual time steps. Experiments showed that Dynamic Convolution Network got better performance and decoded faster than the original Transformer. This architecture has already been implemented in Fairseq.

2.3 Linear Combination Transformer

For the better use of each layer's output in the Transformer, we proposed a new architecture called Linear Combination. In the original Transformer, each encoder layer only transfers its output to the next layer and the decoder only accepts the output of the final encoder layer. In this condition, some grammar or semantics information may be lost even residual connections are applied in each layer. Therefore, we collect each layer's output and calculate them as the final output of the encoder through a weight-sum function. After this operation, the final output is transferred to the decoder. Additionally, it only increases a few parameters which are the same as the number of encoder layers. The experimental results showed that the linear combination function leads the model to perform better.

[1]https://github.com/pytorch/fairseq

2.4 Transformer with Layer Aggregation

For further research of gaining information of each layers, we used layer aggregation mechanism both in the encoder side and decoder side, iterative deep aggregation for the encoder side, hierarchical deep aggregation for the decoder side (Yu et al., 2018), and the linear operation for the encoder side and decoder side. Hierarchical deep aggregation requires the number of layers to be the power of 2, so the number of layers in decoder was set to be 8. Originally, the Transformer only utilizes the top layer' output of the encoder and decoder, which misses the opportunity to exploit the useful information in other layers. Some recent studies reveal that simultaneously exposing all layer representations performs better for natural language processing tasks (Peters et al., 2018; Shen et al., 2018; Dou et al., 2018). In our experiments, we compared the translation results about whether using layer aggregation or not, and found that models with the layer aggregation performed better.

2.5 Encoder Branches with SE-pre in Transformer

Increasing the width of network can improve the model performance effectively and recent works such as Evolved Transformer (So et al., 2019) have proved this idea. Inspired by this, we proposed a new architecture using multi branches mechanism in the encoder side, self-attention for one branch and depthwise separable convolutions (Kaiser et al., 2017) for the other. The outputs of different branches are aggregated by gating unit or just averaging them. We also tried to use SE-pre method (Hu et al., 2018) to replace residual connection and gained a better performance. To reduce the number of parameters, we shared the parameter of different layers in depthwise separable convolutions. In source side, the model has a stack of 6 layers and each layer contains a self-attention sub-layer, a depthwise separable convolution sbu-layer, a gating unit and a FFN sub-layer. In target side, we used the same structure as vanilla decoder in Transformer. Compared with vanilla Transformer, our novel structure outperformed significantly in EN-ZH translation task.

3 Experiment Techniques

3.1 Back Translation

Since Sennrich et al. proposed a method which can translate target side monolingual corpora into

source side to add synthetic data and exploit large corpora, back translation has become a routine operation to build a state-of-art system in translation tasks. Target-side monolingual data plays an important role in neural machine translation systems, so we investigated the use of monolingual data for NMT. In general, we translated monolingual English sentences into Chinese sentences using our English→Chinese baseline system and translated monolingual Chinese sentences into English sentences using our Chinese→English baseline system. To improve the quality of the synthetic corpus, we also conducted a strict data filter which was also used in data preprocessing to exclude bad sentences with low sentence score.

To select sentences for back-translation, we trained unsupervised neural language models with Transformer architectures on target-side bilingual corpora and used them to score these monolingual sentences. We chose News-Discuss corpora 2017 and News-Discuss corpora 2018 which contained about 0.3B sentences totally as our target-side monolingual corpora in Chinese→English translation tasks. We first selected about 80M English sentences from the target-side monolingual corpus based on language model scores, which reflected their similarity to the in-domain corpus. Then we translated them into Chinese sentences and got about 80M sentence pairs. After that, we trained another translation model with Transformer architecture on original bilingual corpora. To calculate bilingual scores for those synthetic sentence pairs, we used the model to translate source-side synthetic sentences and scored their losses with target-side sentences. Finally, we selected 8M sentence pairs with high LM scores and low translation losses and added them to the original corpus.

For English-Chinese translation task, we used XMU monolingual corpus[2] instead of News-Discuss corpora, because XMU corpus contained more in-domain and higher-quality Chinese-side sentences than other monolingual corpora. All other filter operation was same as Chinese-English translation task. Finally, We got 3M synthetic data adding to original corpus.

3.2 Fine-tuning

The Transfer Learning had been used in the field of Computer Vision for a long time, and it had generated significant results (Razavian et al., 2014; Shelhamer et al., 2017; He et al., 2016; Huang et al., 2017). Recent Researches have shown that transfer learning can be extended to natural language processing (NLP) and reinforcement learning. Several papers have indicated that transfer learning and fine-tuning has achieved great success in NLP. (McCann et al., 2017; Peters et al., 2017, 2018; Howard and Ruder, 2018)

In our work of the WMT19, the News-Commentary-v14 was chosen as the in-domain corpus, and the rest of training dataset and the monolingual back-translation corpus were used as the out-domain corpus. In order to enlarge the in-domain corpus, we exploited the algorithm detailed in Duh et al.; Axelrod et al.. Three methods were used to select sentence pairs from large out-domain corpus that are similar to the in-domain corpus, and these sentence pairs were added into the in-domain corpus. Then these new in-domain corpus we got were used to fine-tune the baseline model by continuing training a few steps. The three methods to select similar sentence pairs in our experiments as follows: the KenLM, the Transformer language model, and the tf-idf algorithm.

N- Language Model: According to the work of Deng et al., the in-domain corpus was set as *I* and the out-domain corpus was set as *O*. A smaller out-domain corpus *o* was got from the out-domain corpus by random sampling, and this corpus has similar size with corpus *I*. Then the KenLM was used to train 3-gram language models on the source side and target side of the corpus I and o respectively (H_{I-src}, H_{I-tgt}, H_{o-src} and H_{o-tgt}). After that, all the sentence pairs s from out-domain corpus O were passed into these language models, and scored by using the bilingual cross-entropy difference:

$$[H_{I-src(s)} - H_{I-tgt(s)}] + [H_{o-src(s)} - H_{o-tgt(s)}]$$

At last, the top 20 sentence pairs with lowest scores were add into the in-domain corpus to fine-tune the translation model.

Transformer Language Model: Similar to the above method, the language model with Transformer architecture from Tensor2tensor[3] was used to train the source side and target side of the corpus I and o respectively. The bilingual cross-entropy difference was used to get top 20 similar sentence

[2]http://nlp.nju.edu.cn/cwmt-wmt/

[3]https://github.com/tensorflow/tensor2tensor

pairs from the out-domain corpus to generate new in-domain corpus.

TF-IDF Algorithm: The tf-idf algorithm was chosen to calculate the similarity of the sentences in the in-domain corpus and out-domain corpus. Then we got top 20 similar sentence pairs from out-domain corpus by using the tf-idf scores.

3.3 Ensemble

Ensemble learning, which trains multiple learners and combines them, is a widely used technique in many real-world tasks. Model ensemble has been successfully applied to neural machine translation system, it combines the full probability distribution over the target vocabulary of different models at each step during sequence prediction. We implemented model ensemble module in Tensor2tensor and Fairseq, obtained an improvement of up to 1.2 bleu over the highest single model result. Noticed that simply increasing the size of an ensemble does not necessarily improve translation performance, and brute-force search of all models is unrealistic. As the number of models increases, the decoding of ensemble will take more time than single model, and exceed the limits of computer resource capacity. So we developed an approach that is capable of verifying model combination fast and effectively.

In our algorithm, all the ensemble models are firstly sorted by performance with beam_size = 4. At the first iteration, we selected the best N models and combined them. While it is known that enlarging beam_size can improve decoding performance, in order to verify model combination speedily, beam_size was chosen as 1. After that, we selected the M best model combinations, and decoding them with beam_size = 4 again to further reduce the combination size. Once the first iteration was finished, we added two or four new models to the existed model combination, and then put them into a standard ensemble process described above in the second iteration. The iteration loop will continue until all the models have joined ensemble process. If the number of models is too large, decoding with CPU can be an alternative. Finally, we chose the optimal model combinations, and then increased beam_size and modified the length penalty to gain better translation performance.

Model and data diversity are important factors for ensemble system, so we trained diverse models depending on different parameters, different model architectures, and different training data sets. In order to boost the ensemble performance, all the models have been fine-tuned. For model ensemble strategy, it seems intuitive to employ NMT ensembles by assigning same weights to different models or simply selecting the maximum output probability distributions. In this competition, we adopted a log-avg model ensemble strategy. Both of the max and avg strategy described above we have tried, there was no better result observed.

3.4 Rerank

Reranking is a technique to improve translation quality by choosing potentially better results from the N-Best list. In order to avoid an N-Best list with too many noises, we used strong ensemble systems to generate it. We got an N-Best list with a size of 200+. Then we used 30+ models to score the N-Best list. The models details will be described below. These scores make up several features to represent a sentence in an N-Best list. These features we used including:

Word-alignment feature: These features are generated by using fast-align tools[4] to score the N-Best list and their source sentence.

Language model features: These features are generated by using KenLM and neural language model to score the N-Best list.

Translation models features: Translation model can generate sentences from left to right (L2R) and right to left (R2L), and both source to target (S2T) and target to source (T2S) models can be used to get features. Therefore, there are four kinds (S2T-L2R, T2S-L2R, S2T-R2L, T2S-R2L) of translation model features. In order to get features that can represent the N-Best list more comprehensively, we used translation models that trained with three kinds of frameworks (Tensor2tensor, Fairseq and Sockeye[5]) to generate features.

After getting these features, K-batched MIRA algorithm(Cherry and Foster, 2012) which was implemented in Moses was introduced to the development dataset to get a set of weights. At last, we used these weights to rescore the N-Best list and got final translation results.

[4]https://github.com/clab/fast_align

[5]https://github.com/awslabs/sockeye

4 Experiments Settings and Results

4.1 Data

The WMT18 English↔Chinese translation task contains 24.22M raw data, and the WMT19 English↔Chinese translation task contains 26.17M raw data. There are three high-quality development set: *newstest2017*, *newsdev2017* and *newstest2018*.

4.2 Pre-processing and Post-processing

Firstly, we tokenized the English sentences by using NLTK[6] toolkit and segmented the Chinese sentences with Pkuseg[7] which was produced by Peking University. As a routine operation, we applied BPE (Sennrich et al., 2016b) using Sentencepiece [8] to enable an open vocabulary which contained about 50k words and subwords. For the data selection, we removed duplications in the training data, and designed a filter to exclude bad sentences according to the sentence score obtained by language models and translation models. The final amount of our training data is about 24M bilingual sentence pairs for EN-ZH tasks, and about 22M bilingual sentence pairs for ZH-EN tasks.

We applied post-processing on the outputs of these translation tasks. For EN-ZH translation task, we normalized the punctuations of outputs through converting the single byte character to double byte character and removed the space between Chinese characters. For ZH-EN translation task, we de-tokenized the outputs by Moses toolkit.

4.3 Training Details

All models were trained on 8 GPUs using floating point 16 precision and gradients accumulating (Ott et al., 2018) to employ a bigger batch size as large as 128 GPUs'. We batched sentence pairs by approximate length, limited the number of input and output tokens per batch to 3584 per GPU and re-shuffled the training corpus between epochs. Each training batch contained approximately 450K source tokens and 450K target tokens. We also applied a cosine learning rate schedule (Kingma and Ba, 2015; Loshchilov and Hutter, 2017) where the learning rate is first linearly warmed up for 10K steps from 10^{-7} to 10^{-3} and then annealed following a cosine rate with a single cycle. During training, the label smoothing was employed with $\epsilon_{ls} = 0.1$ and the dropout rate was set from 0.1 to 0.3 (Hinton et al., 2012; Pereyra et al., 2017). The baseline system was trained for about 25 epochs and saved the last 15 epochs to perform checkpoint averaging. At last, we validated the model every 1000 mini-batches against BLEU on the WMT 17 news translation test set.

System	Newsdev2017	Newstest2018
baseline	35.32	
+Data filtering	36.62	
+Back translation	40.23	42.52
+Model enhancement	40.73	42.98
+fine-tuning	41.33	44.10
+ensemble	41.93	46.10
+rerank	42.20	46.40

Table 1: English→Chinese Systems BLEU results on *newsdev2017* and *newstest2018*. As for *newsdev2017* ensemble step, we only mannually selected two models for ensembling test but for *newstest2018*, we applied our ensemble algorithm on all models.

4.4 English→Chinese Systems

Table 1 shows the English→Chinese translation results on the validation set (WMT18 testset). We reported character-level BLEU scores calculated with Moses *mteval-v13a.pl* script [9]. For the baseline system with data filtering, it gained 1.3 BLEU scores compared to the result without filtering. After applying back translation, a single baseline model can improve by about 3.6 BLEU scores. That means synthetic data plays an important role in the success of our system. When it comes to model enhancement, Table 3 shows that each advanced model architecture got a better performance compared to the baseline model. After applying different combinations of the techniques described in Section 2 and 3, we got 11 systems. Thanks to these varieties of model architectures and different data selection strategies, our ensemble system gained a lot and improved about 2 points in term of BLEU. Then we rescored 200+ n-best lists decoding from different single and ensemble systems and finally achieved an improvement of 0.3 BLEU score.

4.5 Chinese→English Systems

Table 2 shows the Chinese→English translation results on the validation set. All results are re-

[6]https://github.com/nltk/nltk
[7]https://github.com/lancopku/pkuseg-python
[8]https://github.com/google/sentencepiece
[9]https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/mteval-v13a.pl

System	Newstest2017
baseline	
+Data filtering	
+Back translation	26.41
+Model enhancement	27.00
+fine-tuning	28.49
+ensemble	29.62
+rerank	29.92

Table 2: Chinese→English Systems BLEU results on *newstest2017*.

Models	EN-ZH	ZH-EN
	dev17	test17
Baseline model(Transformer)	40.23	26.41
Relative Transformer	40.73	26.60
Dynamic Convolution Networks	40.10	26.51
Linear Combination Transformer	40.70	27.00
Layer Aggregation Transformer	40.73	26.93
SE-pre in Transformer	40.51	26.72

Table 3: BLEU results for different model architectures. For EN-ZH, It represents the results on *newsdev2017* and for ZH-EN, it represents the results on *newstest2017*. All models are trained with synthetic data after back translation.

ported with cased BLEU scores. We followed exactly the same settings with the English→Chinese translation system. In this case, the fine-tuning method brought a substantial improvement about 1.4 BLEU scores, showing the advantages of using high-quality in-domain data. For model enhancement, each model architecture got nearly the same BLEU score improvement. Finally, we applied ensemble and reranking techniques, which provided 1.5 BLEU improvements totally over the best single model.

5 Conclusion

We present our NMT systems for WMT19 Chinese↔English news translation tasks. For both translation directions, our final systems achieved substantial improvements up by $4 \sim 5$ BLEU score over baseline systems by integrating the following technique:

1. Data filtering and model enhancements
2. Back translate the target monolingual data set
3. Fine-tuning with in-domain data
4. System combination and reranking.

As a result, our submitted Chinese→English system achieved the second highest cased BLEU score among all 15 submitted constrained systems and our English→Chinese system ranked the second out of 12 submitted systems.

References

Karim Ahmed, Nitish Shirish Keskar, and Richard Socher. 2017. Weighted transformer network for machine translation. *CoRR*, abs/1711.02132.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 355–362.

Colin Cherry and George F. Foster. 2012. Batch tuning strategies for statistical machine translation. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 3-8, 2012, Montréal, Canada*, pages 427–436.

Yongchao Deng, Shanbo Cheng, Jun Lu, Kai Song, Jingang Wang, Shenglan Wu, Liang Yao, Guchun Zhang, Haibo Zhang, Pei Zhang, Changfeng Zhu, and Boxing Chen. 2018. Alibaba's neural machine translation systems for WMT18. In *WMT (shared task)*, pages 368–376. Association for Computational Linguistics.

Zi-Yi Dou, Zhaopeng Tu, Xing Wang, Shuming Shi, and Tong Zhang. 2018. Exploiting deep representations for neural machine translation. *arXiv preprint arXiv:1810.10181*.

Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, pages 678–683.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778.

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

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 328–339.

Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269.

Lukasz Kaiser, Aidan N Gomez, and Francois Chollet. 2017. Depthwise separable convolutions for neural machine translation. *arXiv preprint arXiv:1706.03059*.

Diederick P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6297–6308.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers, WMT 2018, Belgium, Brussels, October 31 - November 1, 2018*, pages 1–9.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.

Matthew E. Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1756–1765.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. Cnn features off-the-shelf: An astounding baseline for recognition. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, CVPRW '14, pages 512–519, Washington, DC, USA. IEEE Computer Society.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. 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.

Evan Shelhamer, Jonathan Long, and Trevor Darrell. 2017. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651.

Yanyao Shen, Xu Tan, Di He, Tao Qin, and Tie-Yan Liu. 2018. Dense information flow for neural machine translation. *arXiv preprint arXiv:1806.00722*.

David R. So, Chen Liang, and Quoc V. Le. 2019. The evolved transformer. *CoRR*, abs/1901.11117.

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.

Felix Wu, Angela Fan, Alexei Baevski, Yann N. Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. *CoRR*, abs/1901.10430.

Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. 2018. Deep layer aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2403–2412.