Low-Resource Machine Translation Based on Asynchronous Dynamic Programming

Xiaoning Jia, Hongxu Hou(✉), Nier Wu, Haoran Li, and Xin Chang

College of Computer Science-College of Software, Inner Mongolia University, Hohhot, China
cshhx@imu.edu.cn

Abstract. Reinforcement learning has been proved to be effective in handling low resource machine translation tasks and different sampling methods of reinforcement learning affect the performance of the model. The reward for generating translation is determined by the scalability and iteration of the sampling strategy, so it is difficult for the model to achieve bias-variance trade-off. Therefore, according to the poor ability of the model to analyze the structure of the sequence in low-resource tasks, this paper proposes a neural machine translation model parameter optimization method for asynchronous dynamic programming training strategies. In view of the experience priority situation under the current strategy, each selective sampling experience not only improves the value of the experience state, but also avoids the high computational resource consumption inherent in traditional valuation methods (such as dynamic programming). We verify the Mongolian-Chinese and Uyghur-Chinese tasks on CCMT2019. The result shows that our method has improved the quality of low-resource neural machine translation model compared with general reinforcement learning methods, which fully demonstrates the effectiveness of our method.

Keywords: Low-resource · Machine translation · Asynchronous Dynamic Programming

1 Introduction

Traditional machine translation [1] uses cross entropy to measure the entropy of the generated token and reference token during the translation process. Its training process is usually to maximize the logarithmic likelihood of each token in the target sentence by taking the source sentence and the translated target token as input. This training method is called maximum likelihood estimation (MLE). Although it is easy to infer, the objective function at the token level in the training is not consistent with the evaluation indicators at the sequence level such as BLEU [6] and the model uses a given real sample word as input to predict the next word. However, since there is no real distribution in the inference stage, the model can only generate the whole sequence by predicting one word

S. Li et al. (Eds.): CCL 2021, LNAI 12869, pp. 16–28, 2021.
https://doi.org/10.1007/978-3-030-84186-7_2

at a time based on its own prediction output. The inconsistent methods of the prediction and inference stages are easy to cause the accumulation of errors. In order to solve the problem of inconsistency and accumulation of errors, a Minimum Risk Training (MRT) [9] algorithm is proposed to alleviate the problem. This training algorithm introduced evaluation index as the loss function, which aims to minimize the expected loss of training data and combines sentence-level BLEU into the loss function, so that the maximum likelihood estimation result of the machine translation model can be significantly improved. However, this function is not necessarily differentiable, which hinders the model from applying the BLEU score directly to the training process.

Aiming at the above problems, reinforcement learning (RL) [7] method has been applied to target optimization at the sequence level. This method directly optimises the BLEU value through Reinforcement training, which is a successful attempt of RL in strategy optimization at the sequence level. BR-CSGAN [13] is the first application of adversarial learning training to machine translation. The idea of reinforcement learning is cited in the generative adversarial network, which enables the neural machine translation model to obtain the BLEU value corresponding to each possible sequence by the bootstrap method, and use this as the basis for optimizing the model parameters. In order to stabilize the training, the loss gradient of the reward is obtained by the reinforcement learning strategy through backpropagation, and finally the translation effect is steadily improved, it uses Monte Carlo for sampling and the sampling efficiency is improved to a certain extent. But due to the independent scalability of the Monte Carlo method, the sampled words are relatively independent of each other and the estimated state value does not depend on the previous state value, thereby reducing the learning accuracy.

In a study [12], it is the first time that reinforcement learning methods were applied to machine translation, and reward shaping is used to improve the effectiveness of monolingual data in translation quality. Nevertheless, the training strategy using reinforcement learning is still restricted by the setting of rewards and the efficiency of sample sampling. As mentioned above, the traditional Monte Carlo method has independent malleability. Compared with the Monte Carlo method, the state value of each iteration of the agent training is determined by the previous state value and it is difficult to achieve the bias-variance trade-off. In order to achieve the trade-off, we hope that the model has the characteristics of low bias and low variance. However, the most serious challenge in the field of deep learning is the problem of high variance. The general solutions to solve the high variance problems are as follows: reduce the complexity of the model, reduce dimensionality of the data and denoise, increase the number of samples, use validation sets and regularization methods. Because of the scarcity of low-resource corpus data, our method involves the use of validation sets and regularization methods to reduce variance. That is, use the validation set to observe the generalization of the training parameters and update the new state through the latest K-step state difference. The Dynamic Programming method further solves the problem of correlation between words, but each updated action reward needs to refer to the previous round of experience, so all the learned experience will be

stored. Based on the above analysis, we combined with the Dynamic Programming to propose an asynchronous training strategy suitable for the training of low-resource neural machine translation model. The bias-variance trade-off is achieved by selectively sampling the experience each time through the experience priority.

In this paper, we used a reinforcement learning strategy combined with Asynchronous Dynamic Programming (ADP) to train machine translation models. The Priortised Sweeping (PS) algorithm is used to determine the experience priority. The higher the priority, the higher the priority. Ultimately, we maximize the priority to generate rewards to update the strategy and use the ADP method for sampling. Compared with the Monte Carlo method, we can pay more attention to the estimation Q of non-terminal state. Compared with the general Dynamic Programming method, the K-step update method avoid the calculation disaster and the real-time supervision of model training can also avoid the accumulation of errors. It improves the accuracy rate and effectively alleviates the sparse reward problem and has a higher universal computing capability.

We map the reinforcement learning algorithm to machine translation according to the Markov characteristics of reinforcement learning. Similar to the traditional machine translation model, we input the source language sentence into the translation model to generate the target sentence. Here we use the LSTM [3] and Transformer [10] frameworks of the machine translation model based on soft attention. The second chapter of this article is about the introduction of background knowledge. The third chapter is divided into framework, asynchronous strategy and training. The fourth chapter is the experiment and analysis part, and the fifth chapter is the conclusion.

2 Background

2.1 Neural Machine Translation

The typical Neural machine translation (NMT) [11] model is an encoder-decoder framework based on attention mechanism and is usually using a cross-entropy function for training. The encoder first maps the source sentence to $X = (x_1, ..., x_n)$ vector form and then the decoder converts the set of vectors to generate the corresponding target sentence $Y = (y_1, ..., y_n)$. And mark the words of the target statement one by one. In the time step t, the model maximizes the likelihood of the current time step t by using the source statement and the target statement before the time step t as input in the training stage. The goal of the translation model is to maximize the maximum likelihood function, MLE is usually used to optimize the model:

$$L_{MLE} = \sum_{i=1}^{S} logp(\hat{y}^i|x^i) = \sum_{i=1}^{S}\sum_{t=1}^{m} logp(\hat{y}_t^i|\hat{y}_1^i...\hat{y}_{t-1}^i, x^i) \tag{1}$$

where S represents the number of sentences in the training set, m is the length of sentence $\hat{y}^i$.

Among all the frameworks based on the encoder-decoder model, the recently popular Transformer architecture has achieved the best translation quality so far. The main difference between Transformer and the previous RNNSearch [2] or Conv2Seq [4] is that it completely relies on the self-attention mechanism [14], using its own structure to calculate the representation of the source and target sentences without recursion or convolution operations.

2.2 RL Based NMT

Reinforcement learning solves the decision-making problem in complex state space by mapping the environment state to the action process to obtain the maximum cumulative reward and provides a new idea for natural language processing tasks. It reduces the gap between NMT training and inference by directly optimizing evaluation indicators (such as BLEU) during training. Specifically, the NMT model can be regarded as an agent interacting with the environment and the parameter definition strategy is the conditional probability $p\left(y_t|x, y_{<t}\right)$. The agent will select a word from the vocabulary according to the strategy, that is, a candidate word. Once the agent generates a complete sequence, it will get a final reward.Reward is introduced to encourage the model to produce an output with a higher reward. In practice, the reward in machine translation is the BLEU value. Obtaining a higher reward means having a higher BLEU score. Here we denote the sentence-level reward as R, where $\hat{y}$ represents the prediction result of the model, y^i represents the reference translation of sentence i, $R\left(\hat{y}, y^i\right)$ represents the reward value of sentence i and by sampling y from the strategy $p\left(\hat{y}|x^i\right)$. At this time, the goal of reinforcement learning training is to maximize the expected reward:

$$L_{RL} = \sum_{i=1}^{S} R\left(\hat{y}^i, y^i\right), \hat{y}^i \sim p\left(\hat{y}|x^i\right), \forall_i \in [S]. \tag{2}$$

3 Approach

In this section, we describe in detail the asynchronous strategy of NMT model training, apply the reinforcement learning algorithm to the neural machine translation model, use the ADP strategy to train the gradient and update the network with parameterization.

3.1 Priority Acquisition of Experience

We abstract machine translation into a Markov process, where s represents the network hidden layer of a specific state of the model, a represents the model decoding prediction, P represents the state transition probability, *reward* represents the BLEU value reward that can be obtained when transferring from one state s to another state s', *strategy* π represents the probability distribution of any action that the agent may take next in a state s, it can also be

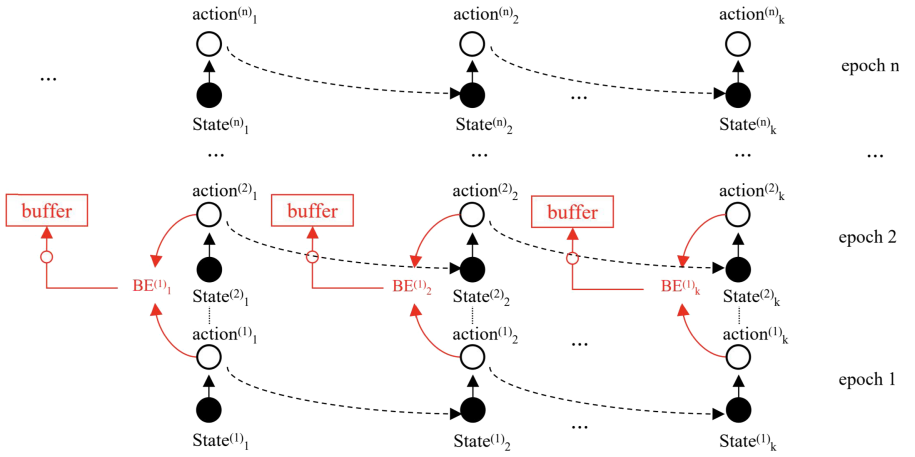


Fig. 1. The illustration of the experience priority.

said that our strategy is to directly select a certain action next in each state. In any case, the reinforcement learning method aims to learn an efficient strategy from the environment to enable agents to have certain model generalization. We can continuously improve our strategy so that we can finally get the maximum cumulative reward. In order to ensure the consistency and rationality of sampling in the training and inference stages and to avoid the problem of strictly traversing each state in Dynamic Programming. Different from the previous work, this paper uses an ADP to sampling, namely the important state priority update(PS) algorithm. The algorithm learns strategy during sampling and reinforcement training. Bellman error is used to calculate the experience priority based on the experience learned from the strategy. The experience that will be collected in the next round of training is determined by priority.

As shown in formula (3), we use the idea of Bellman Error to determine the experience priority situation and use the absolute value of the difference between the state value of the previous round and the current state value to judge. If the Bellman Error is large, it means that the state is unstable and it is necessary to give priority to updating. The experience priority is shown in Fig. 1. We use translation prediction probability *action* instead of state value to judge and record the Bellman Error of each *action* in the first round of epoch and the second round of epoch as $BE^{(1)}_{1\ldots k}$, and so on. After that, $BE^{(1)}_{1\ldots k}$ is stored in the buffer for the optimized call of the asynchronous strategy later. Compared with the sumtree method to extract data priority, the difference data obtained by our method is more dependent on the information content of the previous stage and has more correlation between contexts.

$$\text{Bellman Error} = |\max_{a\in A}(R^a_s + \gamma \sum_{s'\in S} P^a_{ss'} v(s') - v(s))| \tag{3}$$

We store the sorted experience priorities in a specific experience pool, so that the subsequent asynchronous training strategy can extract experience from the experience pool in turn to optimize the model. This method is called experience playback technology. Experience playback is to store the experience of past agents in an experience playback pool and then repeatedly sample from the experience playback pool to optimize the strategy. By using past experience many times to train the current strategy, in order to improve the sample efficiency and training stability of the reinforcement learning algorithm.

3.2 Asynchronous Strategy

The action space of machine translation tasks based on reinforcement learning is huge and discrete and its size is the whole word list. In the previous algorithm, we will completely store and update all the learned experiences in each iteration, thus requiring more program resources. Therefore, we use the important state priority update algorithm in ADP, and trains the machine translation model through the reinforcement learning method. The experience determined by Bellman Error is stored in the experience pool, and the experience *value* with large error is extracted from the experience pool through the K-step update method for optimization. K-step to select the difference of all states accumulated and $R\left(\sum_k\right)$ as the final reward. According to the reward guidance next state $State^{(2)}_{1,\ldots,k}$ update prediction and update until repeated Bellman Error tends to stabilize. This method not only improves the value of the experience state, but also avoids the problem of high computing resource consumption inherent in traditional Dynamic Programming. As shown in Fig. 2. We verified the effectiveness of ADP in the experimental part.

We know that establishing an appropriate reward mechanism is crucial to obtaining high-quality translation. In the process of machine translation, the NMT system acts as an agent of reinforcement learning, obtains the status information of the current moment through continuous interaction with the environment. After each action prediction probability step, the translation model uses the difference between the translation prediction probability of the previous round and the current time as the empirical priority evaluation standard in the current environment. In the K-step, the optimization is updated to the model convergence by selecting the priority to maximize.

Compared with the general Dynamic Programming method, our method uses ADP method for sampling and uses the K-step update method to select the priority to maximize the update to avoid the consumption of computing resources. Moreover, the real-time supervision of model training also avoids error accumulation, improves the accuracy rate while effectively alleviating the problem of reward sparseness and has higher ubiquitous computing capabilities. Drawing on Bellman Error idea, the sum of rewards for all updates in the K-step is used as the final feedback. The formula is as follows:

$$R\left(\hat{y_k}\right) = \sum_{k=1}^{K} Bellman_{1\ldots k} value \tag{4}$$

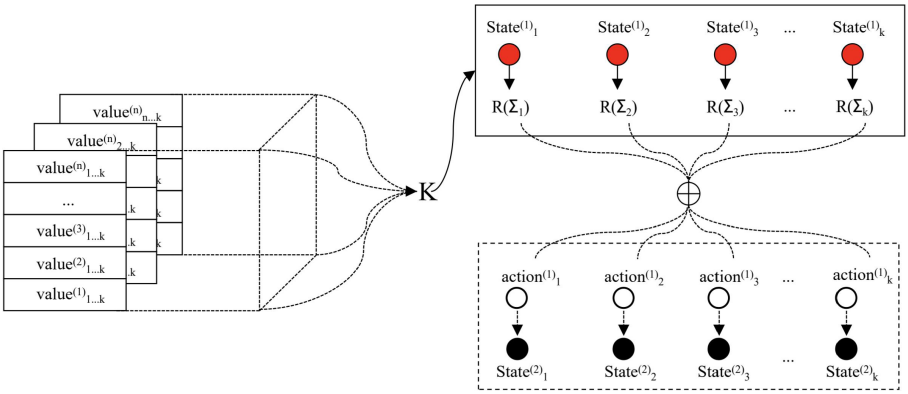


Fig. 2. The illustration of the Asynchronous Dynamic Programming.

3.3 Training

In this paper, ADP strategy is used as a sampling strategy for low-resource reinforcement learning to improve the translation model. The important state with higher experience priority is iteratively updated within K-step, the operation is selected by using the current strategy and the reward is determined by comparing the value of the last moment with the current moment. In the training process, the best sequence comes from the distribution of standard translations. The goal of the training is to find the model parameters that maximize the expected reward, so the objective function is set to:

$$L_K = \sum_{i=1}^{S} R\left(\hat{y}_k^i, y^i\right), \hat{y}_k^i \sim p\left(\hat{y}|x^i\right), \forall_i \in [S]. \tag{5}$$

Finally, in order to further stabilize the RL training process and alleviate the large variance caused by reinforcement learning, we combine the MLE training target with the RL target. The cross-entropy loss function of traditional machine translation is retained in the loss function and it is linearly combined with the reinforcement learning training target. The loss function after mixing is:

$$L_{COM} = \lambda \times L_{MLE} + (1 - \lambda) \times L_K \tag{6}$$

where λ is the hyperparameter to control the balance between MLE and RL. L_{COM} is the strategy to stabilize RL training progress.

4 Experiment and Analysis

We verified the effectiveness of our method on two low-resource NMT tasks, including Mongolian-Chinese (Mo-Zh) and Uyghur-Chinese (Ug-Zh).

4.1 Datasets and Preprocessing

For the Mongolian-Chinese (Mo-Zh) experiment, the data resources used to train the translation model and the translation quality evaluation model come from the Mongolian Intelligent Information Processing Key Laboratory of the Inner Mongolia Autonomous Region. Nearly 250,000 bilingual parallel corpora provided by CCMT2019 are used to train translation models. And 1,000 parallel corpora in CWMT2017 are used as the test set and 1,000 as the verification set. For Uyghur-Chinese (Ug-Zh), 330,000 Uyghur-Chinese machine translation data published by CWMT2017 are used for experiments and 1,000 pieces of data were selected as the test set and the verification set. To avoid the model allocating too much training time to long sentences, all sentence pairs that are longer than 50 at the source or target end will be discarded.

The Stanford Word Segmentation Tool[1] is used to segment the words in Chinese. Because Mongolian and Uyghur are low-resource languages and too many low-frequency words in the vocabulary. So Byte Pair Encoder(BPE) [8] is used to process to alleviate the data sparse problem of low-resource languages to a certain extent in this paper.

4.2 Setting

Our model is improved based on the self-attention Transformer[2] model, use the traditional LSTM[3] and the latest Transformer model as the baseline model. For LSTM, we follow the basic settings and embed the word with a dimension of 512, the learning rate to 0.1, and the beam search size to 4. The settings for Transformer are exactly the same as the original paper, dropout = 0.1, word embedding dimension is 512, header count is 8, encoder and decoder both have 6-layer stack. The quality evaluation is based on the sentence level evaluation standard BLEU. The gradient optimization algorithm uses Adam [5]. During reinforcement learning training, MLE model is used to initialize parameter. The learning rate is set to 0.0001 and the beam search width is set to 6.

4.3 Main Results and Analysis

Comparison Results of Benchmark Experiments. In order to verify the translation performance of the model, we conduct low-resource machine translation comparison experiments with several common representative neural machine translation models under the same hardware conditions and corpus scale. Calculating the BLEU value of

Table 1. The BLEU scores of different NMT systems on Mongolian-Chinese and Uyghur-Chinese.

Model	Mo-Zh	Ug-Zh
Transformer	27.58	36.83
Transformer+RL	31.37	39.71
Our Method	**34.52**	**44.35**
LSTM	25.28	35.67
LSTM+RL	30.95	38.00
Our Method	**33.27**	**42.97**

[1] https://nlp.stanford.edu/software/segmenter.html.
[2] https://github.com/tensorflow/tensor2tensor.
[3] https://github.com/xwgeng/RNNSearch.

Table 2. Asynchronous strategy performance verification.

Priority	Strategy	BLEU		Latency
		Mo-Zh	Ug-Zh	
Random	MC	-	-	-
	DP	-	-	-
	ADP	26.18	33.82	15.7 ms
KD-tree	MC	-	-	-
	DP	-	-	-
	ADP	33.47	42.05	21.3 ms
Bellman	MC	34.36	43.86	382 ms
	DP	34.21	43.75	401 ms
	ADP	**34.52**	**44.35**	**20.5 ms**

each model on the test set separately. It can be seen from the table that our method can surpass the baseline model in both Mongolian-Chinese and Uyghur-Chinese translation tasks. As shown in Table 1.

As can be seen from the translation results, the baseline method of Mongolian-Chinese Machine Translation Transformer is 27.58, Transformer+RL[4] translation method is 31.37, up 3.79. The LSTM baseline method is 25.28, the LSTM+RL[5] is 30.95, and the addition of RL is also 5.67 higher than the baseline, indicating that the introduction of reinforcement learning plays an important role in improving the accuracy of machine translation. Our method is 3.15 BLEU and 2.32 BLEU higher than Transformer+RL and LSTM+RL, respectively. Compared with general reinforcement learning method, our method alleviates the problem that its training strategy is restricted by reward setting and sample sampling efficiency. We avoid computational disasters through the asynchronous update method and the real-time supervision of model training also avoids the accumulation of errors, which can effectively alleviate the problem of sparse rewards while improving the accuracy. The same conclusion applies to Uyghur-Chinese translation.

Asynchronous Strategy Performance Verification. We compare the three priority judgment methods (Random, KD-tree and Bellman error) through different training strategies to verify the effectiveness of the asynchronous strategy and use the latency of each round as the performance verification basis. It can be seen from Table 2 that the priority translation effect of Bellman Error is optimal. Although the translation performance of Monte Carlo and Dynamic Programming strategies are similar to that of ADP, according to the comparison of the duration of each round, it can be found that ADP is much less time-consuming than other strategies.

[4] https://github.com/apeterswu/RL4NMT.
[5] https://github.com/facebookarchive/MIXER.

Due to the nature of asynchronous update based on experience priority, there is no need to wait for the update after each round of traversal is completed. Thereby, reducing the consumption of computing resources, which proves the effectiveness of our method.

Translation Analysis Example. In this paper, Mongolian-Chinese and Uygur-Chinese machine translation examples are explained and the Transformer + RL translation model with better translation performance is used as a comparative example.

From the Fig. 3, we can see that our method is closer to the reference translation than the traditional Transformer+RL translation model. And the translation obtained on the two low-resource corpora of Mongolian-Chinese and Uyghur-Chinese is relatively accurate. The fluency and fidelity of the translation conform to the target language specification, and the quality of the translation is significantly better than its baseline model, which proves that our method can

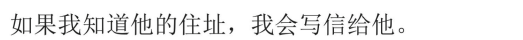
	Mo-Zh
Source	
Reference	如果我知道他的住址，我会写信给他。
Transformer+RL	我知道住址，写信给他。
Ours	**如果**我知道他的住址，我**就**会写信给他。
	Ug-Zh
Source	چىدىمىغۇدەكمەن ، ئېغىرلىقىن بەك بېسىمى خىزمەت .
Reference	工作压力太大，我真受不了了。
Transformer+RL	压力大，我受不了了。
Ours	**工作**压力太大，令**我难以接受**。

Fig. 3. The example of translation results.

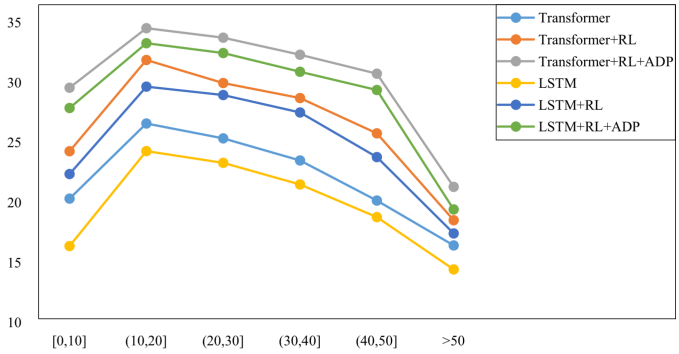


Fig. 4. The length of source sentence.

effectively improve the performance of the low-resource neural machine translation model.

Results of Sentence Length Analysis. We conduct a set of typical experiments using Transformer on the Mongolian-Chinese task to verify the performance of this method on long sentences. We divided the development set data and test set data of the Mongolian-Chinese task according to the sentence length. Figure 4 shows the BLEU scores for different sentence lengths. No matter on LSTM, Transformer or compared to joining RL with the best baseline performance, our work have outstanding behaviors continuously. It is due to our method trains machine translation models based on ADP strategies and increases the ability to analyze low-resource sentence structures through parameter optimization, which makes our method perform better on both long and short sentences.

5 Conclusion

Since different reinforcement learning sampling methods affect the performance of the model and in order to alleviate the quality problem of generated translations in low-resource machine translation tasks, this paper proposes an Asynchronous Dynamic sampling strategy based on reinforcement learning. This strategy selectively samples and stores each time according to the experience priority situation, thereby reducing the consumption of computing resources and improving the value of the experience state. Experimental results show that this method can effectively improve the performance of low-resource machine translation. In the next step, we will combine more low-resource language features to explore more suitable for low-resource machine translation, so as to further improve the performance of low-resource machine translation systems.

References

1. Ahmadnia, B., Dorr, B.J.: Enhancing phrase-based statistical machine translation by learning phrase representations using long short-term memory network. In: Mitkov, R., Angelova, G. (eds.) Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2019, Varna, Bulgaria, 2–4 September, 2019, pp. 25–32. INCOMA Ltd. (2019). https://doi.org/10.26615/978-954-452-056-4_004
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May, 2015, Conference Track Proceedings (2015). http://arxiv.org/abs/1409.0473
3. Cui, Y., Wang, S., Li, J.: LSTM neural reordering feature for statistical machine translation. In: Knight, K., Nenkova, A., Rambow, O. (eds.) NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, 12–17 June, 2016. pp. 977–982. The Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/n16-1112

4. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017. Proceedings of Machine Learning Research, vol. 70, pp. 1243–1252. PMLR (2017). http://proceedings.mlr.press/v70/gehring17a.html
5. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May, 2015, Conference Track Proceedings (2015). http://arxiv.org/abs/1412.6980
6. Papineni, K., Roukos, S., Ward, T., Zhu, W.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, 6–12 July, 2002, Philadelphia, PA, USA, pp. 311–318. ACL (2002). https://doi.org/10.3115/1073083.1073135, https://www.aclweb.org/anthology/P02-1040/
7. Ranzato, M., Chopra, S., Auli, M., Zaremba, W.: Sequence level training with recurrent neural networks. In: Bengio, Y., LeCun, Y. (eds.) 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2–4 May, 2016, Conference Track Proceedings (2016). http://arxiv.org/abs/1511.06732
8. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, 7–12 August, 2016, Berlin, Germany, Volume 1: Long Papers. The Association for Computer Linguistics (2016). https://doi.org/10.18653/v1/p16-1162
9. Shen, S., et al.: Minimum risk training for neural machine translation. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, 7–12 August, 2016, Berlin, Germany, Volume 1: Long Papers. The Association for Computer Linguistics (2016). https://doi.org/10.18653/v1/p16-1159
10. Vaswani, A., et al.: Attention is all you need. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 December, 2017, Long Beach, CA, USA, pp. 5998–6008 (2017). https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
11. Wu, L., et al.: Beyond error propagation in neural machine translation: Characteristics of language also matter. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October – 4 November, 2018, pp. 3602–3611 (2018). https://www.aclweb.org/anthology/D18-1396/
12. Wu, L., Tian, F., Qin, T., Lai, J., Liu, T.: A study of reinforcement learning for neural machine translation. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October – 4 November, 2018, pp. 3612–3621. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/d18-1397

13. Yang, Z., Chen, W., Wang, F., Xu, B.: Improving neural machine translation with conditional sequence generative adversarial nets. In: Walker, M.A., Ji, H., Stent, A. (eds.) 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, 1–6 June, 2018, Volume 1 (Long Papers), pp. 1346–1355. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/n18-1122
14. Yun, H., Hwang, Y., Jung, K.: Improving context-aware neural machine translation using self-attentive sentence embedding. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, 7–12 February, 2020, pp. 9498–9506. AAAI Press (2020). https://aaai.org/ojs/index.php/AAAI/article/view/6494