Improving Non-Autoregressive Machine Translation via Autoregressive Training

Shuheng Wang¹, Shumin Shi²*, Heyan Huang² and Wei Zhang³

¹ School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, Jiangsu, 210094, China
²School of Computer Science and Technology, Beijing Institute of Technology, Beijing, Beijing, 100081, China
³Pingan Life Insurance Company of China, Shenzhen, GuangDong, 518000, China

*Corresponding author’s e-mail: bjssm@bit.edu.cn

Abstract. In recent years, non-autoregressive machine translation has attracted many researchers’ attentions. Non-autoregressive translation (NAT) achieves faster decoding speed at the cost of translation accuracy compared with autoregressive translation (AT). Since NAT and AT models have similar architecture, a natural idea is to use AT task assisting NAT task. Previous works use curriculum learning or distillation to improve the performance of NAT model. However, they are complex to follow and difficult to be integrated into some new works. So in this paper, to make it easy, we introduce a multi-task framework to improve the performance of NAT task. Specially, we use a fully shared encoder-decoder network to train NAT task and AT task simultaneously. To evaluate the performance of our model, we conduct experiments on several benchmark tasks, including WMT14 EN-DE, WMT16 EN-RO and IWSLT14 DE-EN. The experimental results demonstrate that our model achieves improvements but still keeps simple.

1. Introduction

Neural machine translation (NMT) has achieved a great success in recent years [1,2,3]. According to diverse types of inference, NMT can be divided into two types: Autoregressive translation (AT) and Non-autoregressive translation (NAT). AT models generate the target tokens one-by-one from left to right [1,2]. This decoding method simplifies the translation process and achieves state-of-the-art performance, but it is often censured for its high decoding latency [4]. In contrast, NAT models significantly speedup the inference stage, which simultaneously generate the target tokens [4,5,6]. However, NAT models suffer from the low translation accuracy.

In recent years, many works have been done to explore the relations between NAT models and AT models. For examples, Guo [7] used curriculum learning to finetune AT model to transfer knowledge from AT model to NAT model. And Liu [8] similarly used curriculum learning to explore the task-level transferring from AT model to NAT model. While these works improve the performance of NAT models, these works are complex and difficult to be integrated into some recent works. Meantime, Hao [9] used a NAT-AT multi-task framework with shared encoder to strengthen capability of encoder. Their method is easy to follow, but the number of parameters in their model is more than vanilla NAT model.

Inspired by these works, in this paper, we proposed a fully shared multi-task NAT-AT model. Different from [9], we use a fully shared encoder-decoder framework for NAT task and AT task,
where AT task can be seen as the auxiliary task. In this way, the linguistic knowledge learned by AT task can directly guide the learning process of NAT model. To verify the performance of our proposed method, we conduct experiments on WMT14 EN-DE, WMT16 EN-RO and IWSLT14 DE-EN tasks. And the results demonstrate that our model can significantly improve the performance of vanilla NAT model. Meantime, to proved that our model can be easily migrated to recent works, we also implement our method on DCRF-NAT [10].

2. Related Works
In this section, we will describe our related works. Firstly, we will introduce the related works about NAT models. And then, we will describe the multi-task learning used by us.

2.1. Non-autoregressive Translation
Typically, given a source sentence \( X = \{x_1, x_2, \ldots, x_n\} \), autoregressive machine translation maps the source sentence into hidden representations \( h_X \). Based on \( h_X \), AT model generates the target sequence \( Y = \{y_1, y_2, \ldots, y_n\} \) word by word from left to right [1,2,3]. While AT model is easy and achieves state-of-the-art performance, it suffers from the high inference latency. To reduce the decoding latency, NAT model has been introduced [4]. Different from the AT model, NAT model generates target sequence in one-shot, which can significantly reduce the inference latency. But the translation accuracy of NAT model is still worse than AT model. Recently, some works has been introduced to improve the performance of NAT models [5,6,10]. Meantime, some researchers begin to study the relations between the NAT model and AT model and have introduce some works based on curriculum learning to shift the knowledge from AT model to NAT model [7,8,9,17]. While these methods achieve some improvements, they are too complex to be integrated into some recent works. Different from that, in this work, we introduce a multi-task fully shared encoder-decoder framework to training NAT model and AT model.

2.2. Multi-Task Learning
Multi-task learning [11] has been widely used in NLP tasks. And in recent years, multi-task learning also has been adapted in AT tasks. For example, Garg [12] used multi-task learning to generate the target sequence and extracted the alignments between the source sentence and target sentence. In this work, based on multi-task learning, we will learn NAT model and AT model simultaneously.

Noted that, our work is very related to [9]. The key difference is that they only use a shared encoder, but we use a fully shared encoder-decoder network. They use AT task to strengthen the capability of encoder. Besides that, our work use AT task to help the NAT model learn the sequence information. In additional, the number of parameters used in [9] is more than our method.

3. Method
In this section, we will describe the method used in our work.

3.1. Model Architecture
The model architecture used in our model is identical with the vanilla model [4]. And the model consists of two sub networks: encoder and decoder. The parameters used in encoder and decoder are
shared between NAT model and AT model. Due to the fully shared parameters, the model, itself, cannot distinguish which is NAT task or AT task. So, we use different mask to help the model distinguish NAT task and AT task. As shown in the Figure 1, the yellow color denotes “1” and white denotes “0”. The left figure is a causal mask and used for AT task to learn target sequence from left to right. And the right figure denotes bidirectional mask, and the model can see all words, which is used for NAT task. The NAT task and AT tasks are trained simultaneously by multi-task framework described in next section.

3.2. Multi-Task Optimizing

To train our proposed method, we define the loss function of our method $\mathcal{L}$ as follows:

$$\mathcal{L} = \mathcal{L}_{\text{NAT}} + \lambda \mathcal{L}_{\text{AT}}$$  \hspace{1cm} (1)

where, $\mathcal{L}_{\text{NAT}}$ is the loss function of NAT task. Similarly, $\mathcal{L}_{\text{AT}}$ is the loss function of AT task. The final loss function is the addition of $\mathcal{L}_{\text{AT}}$ and $\mathcal{L}_{\text{NAT}}$. $\lambda$ is a hyper-parameter to balance the performance of NAT task and AT task. Noted that $\lambda$ should be less than 1. If $\lambda$ is greater than 1, the model may not be converged.

3.3. Training and Inference

During training, given a bilingual sentence pair $(X, Y)$, we feed the sentence pair into NAT model and AT model separately. Then, we calculate the loss of NAT model and AT model separately, and get the final loss by the Equation. 1. Due to the fully shared parameters, the model finally can learn the sentence distribution from NAT task and AT task. During inference, we only focus on the NAT model, and throw AT model aside. So, the decoding speed of our model is equal to the vanilla NAT model.

| Model                              | WMT14 EN-DE | WMT16 EN-RO | IWSLT14 DE-EN |
|------------------------------------|-------------|-------------|---------------|
| Autoregressive Models              |             |             |               |
| Transformer [1]                    | 27.30       | 34.16       | 32.99         |
| Non-Autoregressive Models          |             |             |               |
| Vanilla-NAT [4]                    | 17.69       | 26.22       | /             |
| Vanilla-NAT (NPD100)               | 19.17       | 29.79       | 24.21         |
| FCL-NAT [7]                        | 21.70       | /           | 26.62         |
| TCL-NAT [8]                        | 21.94       | /           | 28.16         |
| Hint-NAT [17]                      | 21.11       | /           | 25.55         |
| Multi-Task-Shared-Encoder [9]      | 21.05**     | /           | /             |
| Ours                               |             |             |               |
| Multi-Task-Full-Shared             | 21.98       | 27.37       | 29.04         |

4. Experiments

To evaluate the performance of our method, we compare the performance of our method with our models using AT task as assistance. In this section, we will describe the settings and results of our experiments.

4.1. Datasets and Settings

**Datasets:** To evaluate the performance of our model, we conduct experiments on three benchmark translation tasks: WMT 14 EN-DE, WMT16 EN-RO and IWSLT 14 DE-EN. They consist of 450M, 610K and 160K bilingual sentence pairs, respectively. After tokenizing the sentence into words with

1 In this work, we don’t compare the performance of our model with NPD. And the results of other models except vanilla NAT model also are reported without NPD.
script provided by Moses\(^2\). We tokenize the word into sub-unit words using BPE [13]. For WMT tasks, we use a vocabulary with size 32k shared by the source and target language. For IWSLT task, we set the vocabulary size as 10k.

**Settings:** Following the previous works [4,7,8], we use Transformer-small (n\_layer=5, n\_embed=256, n\_hidden=512, and n\_head=4) for IWSLT14 task. For WMT task, we use Transformer-base (n\_layer=6, n\_embed=512, n\_hidden=2048, and n\_head=8) as the settings. We use Adam [14] as our optimizer to train all models. We train our models on 8/1 Nvidia Tesla V100 GPUs with batch size 8000 for WMT tasks and IWSLT task respectively. We train the model for 300k steps and create the final model by averaging the 5 best checkpoints. Due to the critical importance of sequence-level distillation [4], we also use distilled corpus as the train sets. In additional, we use BLEU [15] as the metric to evaluate the performance of our model.

4.2. **Main Results**

We show the main results of our model in the Table 1. Compared with vanilla NAT model, our model achieves significant improvements by a large margin. Especially, compared with vanilla NAT model, our model obtains 5.0+ and 1.0+ bleu scores on WMT 16 EN-RO and IWSLT 14 DE-EN tasks, which can demonstrate that with the assistance of AT task, NAT model can obtain better performance. Although our method achieves comparable results with FCL-NAT and TCL-NAT, our method is easier to implement and be incorporated into recent works.

In additional, our model has the same architecture with vanilla NAT model. And the autoregressive feature mainly lies on the causal mask. So, in this way, the decoding latency will not be affected and will be consistent with vanilla NAT model.

Compared Multi-Task-Shared-Encoder, which is related to us, what should be noted is that Multi-Task-Shared-Encoder only shares the encoder and adds an external decoder for AT task. So, during training, Multi-Task-Shared-Encoder will increase the numbers of parameters. Our model does not increase the number of parameters but achieves better performance.

4.3. **Incorporating Into Recent Work**

Compared with curriculum learning based model, our model can be easily incorporated into recent works. And in this work, we also fuse our model with DCRF-NAT [10]. Meantime, to evaluate our model with DCRF, we conduct experiments on IWSLT 14 DE-EN task. And we show the results on the Table 2.

| Model                     | BLEU  |
|---------------------------|-------|
| Vanilla NAT (NPD100)      | 24.21 |
| DCRF-NAT [10]             | 27.44 |
| Multi-Task-Fully-Shared   | 29.04 |
| +DCRF                     | 29.58 |

From Table 2, we can observe that our method can be easily incorporated into recent works and can continue to improve the performance. We can see that our model with the assistance of AT task can obtain better result than DCRF-NAT. If adding DCRF on our method, our method can achieve about 2.0+ BLEU scores than DCRF-NAT, which demonstrate the capacity of our method.

4.4. **Effect of Sequence Length**

In this section, we analysis the impact of our method on different target length. We divide the sentences according to their length, and calculate the performance of our method on them. We show the results on the Figure 2.

\(^2\) [https://github.com/moses-smt/mosesdecoder](https://github.com/moses-smt/mosesdecoder)
From Figure 2, we can see that our method can improve the performance on various length. Although the performance of our method and vanilla NAT achieves similar performance on the bucket length less than 10, our method achieves better results when the length is greater than 10. Compared with vanilla NAT, with the assistance of AT task, our method has a better performance oppositely, which exactly proves that autoregressive training can improve the performance of non-autoregressive machine translation.

Figure 2: The performance of different length buckets on IWSLT14 DE-EN valid test

4.5. Effect on Repetitive words
In the previous works [4,16], the issue of repetitive words is a huge important factor which significantly reduces the performance of NAT models. So, in this section, we conduct an analysis of effect of method on reducing the repetitive words on IWSLT 14 DE-EN validation set. And we show the results on the Table 3.

Table 3: The average repetitive words on IWSLT 14 DE-EN validation set

|         | Vanilla-NAT[1] | NAT-REG[16] | Mask-Predict[6] | Ours |
|---------|----------------|-------------|-----------------|------|
|         | 2.30           | 0.90        | 0.48            | 0.02 |

From Table 3, our method can significantly reduce the number of repetitive words compared to vanilla NAT model. Our method even generates fewer repetitive words than NAT-REG, which adds a regularization to reduce the repetitive words. Even compared with iterative-refine base model Mask-Predict, our model still achieves a better performance on reducing the number of repetitive words. The results also demonstrate the performance of NAT model can be improved by the assisting of AT model.

5. Conclusion
In this work, to improve the performance of NAT, we proposed a multi-task fully shared framework to improve the performance of NAT models. Especially, we use a fully shared encoder-decoder for AT task and NAT task, and multi-task training to train AT model and NAT model synchronously. During training, the knowledge learned by AT model can effectively assist the learning process of NAT model. We conduct experiments on three benchmark tasks, and the results demonstrate that with the assistance of autoregressive training, the performance of non-autoregressive model can be significantly improved. To exploring the expandability of our model, we will incorporate more recent works into our model for future work.
Acknowledgments
We would like to thank the reviewers for their insightful comments. And our work is supported by the National Science Foundation of China (61732005, 61671064).

References
[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).
[2] Gehring, J., Auli, M., Grangier, D., Yarats, D., & Dauphin, Y. N. (2017, July). Convolutional sequence to sequence learning. In International Conference on Machine Learning (pp. 1243-1252). PMLR.
[3] Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
[4] Gu, J., Bradbury, J., Xiong, C., Li, V. O., & Socher, R. (2018, February). Non-Autoregressive Neural Machine Translation. In International Conference on Learning Representations.
[5] Guo, J., Tan, X., He, D., Qin, T., Xu, L., & Liu, T. Y. (2019, July). Non-autoregressive neural machine translation with enhanced decoder input. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 3723-3730).
[6] Ghazvininejad, M., Levy, O., Liu, Y., & Zettlemoyer, L. (2019, November). Mask-Predict: Parallel Decoding of Conditional Masked Language Models. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (pp. 6112-6121).
[7] Guo, J., Tan, X., Xu, L., Qin, T., Chen, E., & Liu, T. Y. (2020, April). Fine-tuning by curriculum learning for non-autoregressive neural machine translation. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34, No. 05, pp. 7839-7846).
[8] Liu, J., Ren, Y., Tan, X., Zhang, C., Qin, T., Zhao, Z., & Liu, T. Y. (2020). Task-level curriculum learning for non-autoregressive neural machine translation. arXiv preprint arXiv:2007.08772.
[9] Hao, Y., He, S., Jiao, W., Tu, Z., Lyu, M., & Wang, X. (2020). Multi-task learning with shared encoder for non-autoregressive machine translation. arXiv preprint arXiv:2010.12868.
[10] Sun, Z., Li, Z., Wang, H., Lin, Z., He, D., & Deng, Z. H. (2019). Fast structured decoding for sequence models. arXiv preprint arXiv:1910.11555.
[11] Caruana, R. (1997). Multitask learning. Machine learning, 28(1), 41-75.
[12] Garg, S., Peitz, S., Nallasamy, U., & Paulik, M. (2019). Jointly learning to align and translate with transformer models. arXiv preprint arXiv:1909.02074.
[13] Sennrich, R., Haddow, B., & Birch, A. (2015). Neural machine translation of rare words with subword units. arXiv preprint arXiv:1508.07909.
[14] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
[15] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics (pp. 311-318).
[16] Wang, Y., Tian, F., He, D., Qin, T., Zhai, C., & Liu, T. Y. (2019, July). Non-autoregressive machine translation with auxiliary regularization. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 5377-5384).
[17] Li, Z., Lin, Z., He, D., Tian, F., Qin, T., Wang, L., & Liu, T. Y. (2019). Hint-based training for non-autoregressive machine translation. arXiv preprint arXiv:1909.06708.