

High Training Efficiency Transformer for Multi-scenario Non-Autoregressive Neural Machine Translation

Xiangyu Qu^{1,2}, Guojing Liu³, and Liang Li^{1,2}(✉)

¹ School of Cyber Science and Technology, Shandong University, Qingdao, China

² Key Laboratory of Cryptologic Technology and Information Security of Ministry of Education, Shandong University, Qingdao, China
li.liang@sdu.edu.cn

³ Faculty of Information Science and Engineering, Ocean University of China, Qingdao, China

Abstract. Non-autoregressive neural machine translation (NAT) focuses on improving reasoning efficiency through parallel decoding. However, NAT models training method lack improvement compared with the autoregressive translation (AT) models, which leads to an imbalance between training efficiency and inference speed. In this paper, we propose Padding Accelerated Training (PAT) for NAT. Specifically, we pad short sentences not with padding tokens but with another real training sentence, and apply Sequence Concatenating attention (SC) to obtain the sentence-level blocking matrix to prevent multiple sentences from interfering with each other. Experiments show that PAT is applicable to both sentence-level and document-level machine translation scenarios. While ensuring translation performance, PAT improves training speed by more than 2 times in multiple experimental tasks.

Keywords: Machine translation, Non-autoregressive transformer, High Efficiency Training.

1 Introduction

Nowadays, transformer framework is widely used in machine translation tasks [1]. Vanilla transformer utilizes an autoregressive approach which inevitably results in slow decoding speed. To address this issue, several models based on Non-autoregressive Transformers (NAT) have been proposed with the aim of speeding up inference through parallel decoding [2]. The current mainstream NAT models primarily consist of two approaches. The first approach is iterative NAT transformers [3, 4], which adopt a technique similar to the masked language model (MLM) [5]. In this approach, multiple tokens are generated for each iteration to capture the interdependence of words in the target sentence. However, this iterative refinement method of generating multiple decoding does not yield significant improvements in accelerating reasoning. The second method is single pass parallel decoding (or Non-iterative) of NAT. This approach decodes and outputs all translations simultaneously, leading to a significant acceleration in reasoning. However, due to the assumption of independence, the NAT model fails to accurately learn the dependencies between target markers in the actual data distribution.

As a result, its accuracy under performs the AT model. To address this issue, improvements on the NAT model are mainly based on refining its target [6] or incorporating additional input into the decoder [7]. In [8], the authors reduce the learning space for output tokens in four aspects (training corpus, model architecture, training goals and learning strategies). A recent work [9] proposed a directed acyclic transformer while still introducing additional multi-modal information.

The current research mainly focuses on how the NAT model achieves a balance between inference efficiency and decoding quality, but ignores the improvement of training efficiency. For example, the inference speed of the NAT model is more than ten times that of the AT model, but the training speed is not improved. This leads to an imbalance between training efficiency and inference efficiency.

We analyzed the corpus data and found that the long-tail distribution of sentence lengths in human language often makes the size of padded inputs bounded by an exceptionally long sentence. As a result, most sentences in a training batch are much shorter than the input size, and thus have to be padded with a large number of padding tokens. For example, we found that in WMT14 En→De, the padding tokens account for nearly 70% of the model input under standard training setting, and the padding rate can be even higher as the batch size increases. In this paper, we present Padding Accelerated Training (PAT), a method that eliminates zero padding and replaces it with real sequence padding to significantly increase the proportion of real tokens in the training data by over 2 times. However, simple sequence concatenation may lead to mutual contamination of attention matrices during training. To solve this problem, we introduced a Sequence Concatenating (SC) attention mechanism to obtain blocks of the single sequence attention matrices from the concatenated multi-sequence attention matrix so as to achieve lossless recognition of single sentence attention. At the same time, SC attention mechanism can be transferred from sentence-level translation tasks to document-level translation tasks, enabling the application of the NAT model in multi-scenario translation tasks for the first time. Furthermore, we enhanced the glancing transformer with PAT to expedite decoding.

Experiments on WMT14 En↔De and WMT17 En↔Zh datasets show that PAT improves the training efficiency of the NAT model by more than 2 times, and achieves the best translation quality in multiple translation directions. Experiments on the document-level MT datasets TED, News and Europarl show that PAT can directly apply the NAT model to document-level translation tasks and increase the translation speed by more than 30 times. To the best of our knowledge, we are the first to design a NAT model suitable for multi-scenario translation tasks.

2 Proposed Approach

In this section, we describe our proposed PAT-Transformer in detail. We first introduce the SC attention mechanism applicable to concatenate sequences, and then describe the modified glancing training strategy.

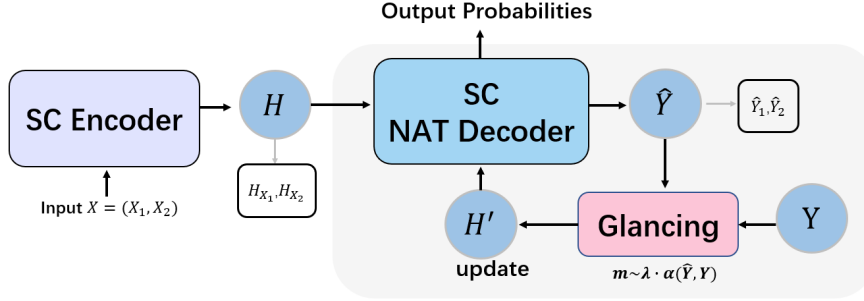


Fig. 1. Glancing training model structure based on PAT. Input X represents a multi-sequence consisting of X_1 and X_2 . H represents the encoder hidden representation, and the Glancing module receives \hat{Y} and ground truth Y , and then adjusts H to H' .

2.1 Padding accelerated training

Sequence concatenation. The purpose of concatenating sequences is to make the sequences within the batch as long as possible to reduce the padding rate. With this guiding principle, we propose an efficient sequence concatenating algorithm, as shown in Algorithm 1.

Algorithm 1: Sequence concatenation

Input: T : number of batches in dataset.
 B : Batch = $(x_1, y_1), \dots, (x_n, y_n)$;
for $s \leftarrow 0$ **to** T **do**
 $X_{\max}, Y_{\max} = B(x_{\max}, y_{\max})$;
 Sort 2D(x, y) in $B \rightarrow x_i > x_j$, if $i < j$;
 for $i \leftarrow 0$ **to** n **do**
 if $x_i + x_j \leq x_{\max}$ and $y_i + y_j \leq y_{\max}$ **then**
 pack $(x_i, y_i) \leftarrow (x_i + x_j, y_i + y_j)$;
 end
 end
end

Our algorithm is based on the concept of bin-packing but considers both source and target attributes. At the start of the algorithm, we identify the longest source and target sequences and use them as bins. We do this to flexibly account for the possibility of the longest source and target sequences belonging to different sentence pairs and to maximize sequence filling.

Attention mechanism. In standard attention mechanism, each sequence is designed to attend exclusively to its own relevant tokens and should not attend to tokens from another sequence. Without modification of the attention structure, a sequence may experience interference from other instances during the attention process. To address this issue, we propose a masking strategy within the attention mechanism to accommodate inputs generated through sequence combinations. The standard attention is:

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

where Q , K , V are called Query (Q), Key (K), and Value (V), which are the head components of the attention.

To accommodate sequence combination, we introduce sequence encoding tokens that contain information from different instances, and the new mask matrix is composed of both sequence encoding tokens and padding tokens. Now, the calculation of attention is:

$$\begin{aligned} & \text{Attn}(Q^+, K^+, V^+) \\ &= \text{softmax}\left(\text{M}\left(\frac{Q^+ K^{+T}}{\sqrt{d_k}}\right)\right)V^+ \\ &= \text{softmax}\left(\text{M}\left(\frac{\overbrace{Q_1 K_1^T, Q_1 K_2^T, \dots, Q_n K_n^T}^{n^2}}{\sqrt{d_k}}\right)\right)V^+ \\ &= \text{softmax}\left(\frac{\overbrace{Q_1 K_1^T, Q_2 K_2^T, \dots, Q_n K_n^T}^n}{\sqrt{d_k}}\right)V^+ \\ &= \text{Attn}(Q_1, K_1, V_1), \dots, \text{Attn}(Q_n, K_n, V_n) \end{aligned} \quad (2)$$

where $Q^+ = (Q_1, \dots, Q_n)$ is the query matrix of combinatorial sequence composed of the 1_{st} sequence to the n_{th} sequence. The key matrix $K^+ = (K_1, \dots, K_n)$ and value matrix $V^+ = (V_1, \dots, V_n)$ are also packed together.

The SC attention needs a new mask different from that of standard attention. The reason for this is that the combined sequence formed from n sequences can be represented by n^2 matrices (e.g., $Q_1 K_1^T, Q_1 K_2^T, \dots, Q_n K_n^T$) which are obtained via dot-products. As shown in Figure 2, self-attention is executed within every diagonal entry, so the SC attention is restricted to n zones of $Q_1 K_1^T, Q_2 K_2^T, \dots, Q_n K_n^T$, thereby preventing any intermixing between sequences.

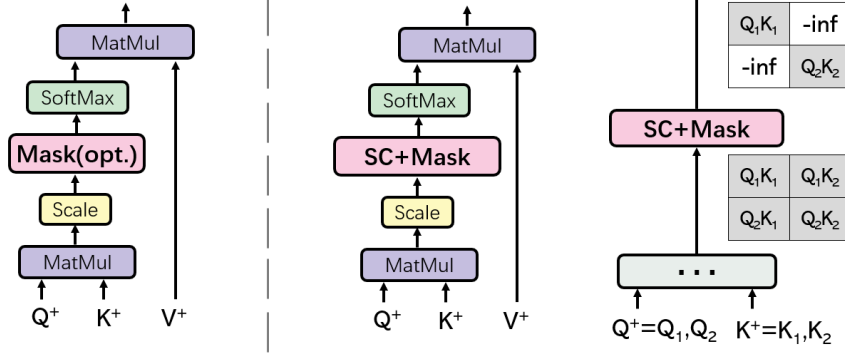


Fig. 2. (left)Mask(opt.) in Scaled Dot-product Attention (middle)Scaled Dot-product Attention with SC. (right)SC mechanism.

2.2 Glancing targets

A typical NAT system uses the conditional independent factorization to maximize the following likelihood:

$$\mathcal{L}_{NAT} = \sum_{t=1}^T \log P(y_t | X; \theta) \quad (3)$$

where θ represents the model parameters. Typically, NAT models leverage the Transformer encoder-decoder architecture for modeling without employing causality in the attention mechanism on the decoder side. However, it has been noted by [2] that in sequence generation tasks like machine translation, where the real data distribution does not typically support the independence assumption, the failure to capture the dependency between target tokens results in a considerable decrease in the performance of NAT. Glancing Transformer (GLAT) [7] is proposed to enhance the ability of NAT models to produce high-quality translations in a single-pass parallel decoding.

As shown in the Figure 1, we adopt glancing training method based on combination sequences, which consists of the following steps:

- Input the combined sequence into the initialization model to obtain the prediction \hat{Y} ;
- Measure the difference between y and target to obtain glancing sample tokens $m \sim \lambda \cdot \alpha(y, \hat{Y})$, where λ is the a hyper-parameter to more flexibly control the number of sample tokens, $\alpha(y, \hat{Y})$ is the difference between the predicted value and the ground-truth, measured by Hamming distance [10];
- Reconstruct the encoder output H and replace it with the corresponding target word embedding using m to obtain H' ;

- Placing H' in the decoder and calculating with the target yields the loss of the remaining tokens.

The objective of the GLAT framework is to reconstruct target tokens that are not directly visible by leveraging contextual information. Unlike masked language modeling [5], GLAT employs an adjustment factor N to dynamically adjust the masking ratio based on the quality of the output results rather than masking a fixed proportion of target tokens.

3 Experiments

3.1 Dataset and Metrics

We verify the effectiveness of our approach on two datasets.

The WMT14 NewsTest task for translating English news articles into German sentences (En \rightarrow De) is one of the most widely used public benchmarks in machine translation. WMT14 contains about 4.5M bilingual corpus, the verification set is *newstest2013*, and the test set is *newstest2014*.

The WMT17 Chinese \rightarrow English (Zh \rightarrow En) task requires translation of news articles written in Chinese to English. The raw WMT17 zh2en training data is comprised of 25 million translation examples from three sources: News Commentary, UN Parallel Corpus and CWMT Corpus.

The TED corpus from IWSLT2017 consists of transcriptions of TED talks, with each talk corresponding to a document. The sentences in the source and target documents are aligned for translation. For testing, we use *test2016-2017*, and for development, we use the remaining documents.

News is a corpus mainly from News Commentary v11, where the sentences are also aligned between the source and target documents. For testing and development sets, it uses *newstest2016* and *newstest2015*, respectively.

Europarl is a corpus extracted from Europarl v7, where the train, development, and tests are randomly split.

To ensure fair comparisons with previous studies, we employ BLEU [11] for WMT14En \leftrightarrow De and WMT17Zh \leftrightarrow En benchmark evaluations. About document-level metrics, we report document-level sacreBLEU(d-BLEU) [12], which is computed by matching n-grams in the whole documents.

3.2 Knowledge Distillation

Knowledge distillation (KD) [2,13] is the most commonly used technology for single pass NAT models. We replace the original target samples with sentences generated by the pre-trained autoregressive transformer, which can help the target model filter noisy samples in the early stage of training. We employ the transformer with the base setting in [1] as the teacher for knowledge distillation.

Table 1. Results on WMT14 En-De and WMT17 Zh-En. Iter is the number of decoding iterations. Speed up is measured on WMT14En-DE test set. Our work and previous research might have employed varying hardware configurations and implementation strategies, making direct comparisons of speed-up unfair.

| Models | Iter | WMT14 | | WMT17 | | Speedup | | |
|--------|-------------------|-------|--------------|--------------|--------------|--------------|-------------|--------------|
| | | EN-DE | DE-EN | EN-ZH | ZH-EN | Training | Inference | |
| AT | Transformer[1] | T | 27.30 | - | - | - | 1.0x | 1.0x |
| | Transformer(ours) | T | 27.86 | 31.55 | 34.82 | 23.99 | 1.0x | 1.0x |
| Iter>1 | DisCo[16] | 4 | 27.34 | 31.31 | 34.63 | 23.83 | 1.0x | 3.5x |
| | SMART[17] | 10 | 27.65 | 31.27 | 34.06 | 23.78 | 1.0x | 2.2x |
| | LevT[3] | 6+ | 27.27 | - | - | - | 1.0x | 4.0x |
| | Mask-Predict[4] | 10 | 27.03 | 30.53 | 33.19 | 23.21 | 1.0x | 1.7x |
| | CMLMC[18] | 10 | 28.37 | 31.41 | - | - | 1.0x | 1.7x |
| Iter=1 | Vanilla-NAT[2] | 1 | 19.69 | 25.63 | 28.36 | 15.06 | 1.0x | 15.5x |
| | CTC[19] | 1 | 16.56 | 18.64 | 26.33 | 11.85 | 1.0x | 14.6x |
| | GLAT[7] | 1 | 25.21 | 29.84 | 32.08 | 21.65 | 1.0x | 15.2x |
| | Ful-NAT[8] | 1 | 27.20 | 31.39 | - | - | 1.0x | 16.8x |
| | ReorderNAT[20] | 1 | 22.79 | 27.28 | - | - | 1.0x | 16.1x |
| | Flowseq[21] | 1 | 27.32 | 28.39 | - | - | 1.0x | - |
| | NAT-DCRF[22] | 1 | 23.44 | 27.22 | 32.18 | 19.98 | 1.0x | 10.4x |
| | PAT+GLAT | 1 | 27.01 | 30.36 | 34.27 | 23.20 | 2.5x | 15.6x |
| | PAT+GLAT+CTC | 1 | 27.55 | 31.39 | 34.22 | 23.51 | 2.5x | 15.5x |

3.3 Baselines

We design our NAT models with the hyperparameters of the base Transformer [1]. For regularization, we set the dropout rate to 0.1 and use Adam optimizer with $\beta = (0.9, 0.999)$. All models are trained for 300k steps using Nvidia A40 GPUs with a batch of 64k tokens. The BLEU scores are evaluated on the validation set after each epoch, and the average of the top 5 checkpoints is taken to obtain the final model. We tune λ for linear annealing from 0.55 to 0.35. All models are implemented in fairseq [14].

3.4 Inference

During inference, the length prediction component allows the model to generate the target sequence by collapsing repeated characters and removing blank symbols from the alignment sequence. In this work, our main focus on Connectionist Temporal Classification (CTC) [15] loss, is able to make accurate predictions even when the input and output sequences have different lengths. By leveraging the length prediction component, it can efficiently find the most likely target sequence given the input sequence, making it a powerful tool for tasks involving variable-length sequences.

3.5 Sentence-level translation results

As shown in Table 1, the performance obtained by our model is in the leading position among all NAT models and significantly improves the decoding speed. The NAT model achieved significant speedup compared to the AT model, but there is still a gap in performance.

Table 2. Results on raw data(Raw) and knowledge distilled data(KD), we are using a test set with a batch size of 1 to evaluate the speedup, "-" indicates that the experimental results are abnormal values, and we have analyzed the reasons below.

| | Models | TED | | News | | TED | | Speedup |
|-----|----------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | | Raw | KD | Raw | KD | Raw | KD | |
| AT | Transformer | - | 26.15 | - | - | 33.38 | 27.11 | 1.0x |
| | +PAT | 27.52 | 27.01 | 27.38 | 26.63 | 34.18 | 33.01 | 1.0x |
| NAT | Vanilla-NAT[2] | - | - | - | - | - | - | 40.8x |
| | CTC[19] | 21.69 | 25.01 | 16.60 | 24.87 | - | 31.72 | 27.1x |
| | GLAT[7] | 19.06 | 25.83 | 11.95 | 22.21 | - | 30.68 | 27.0x |
| | PAT+GLAT | 21.13 | 25.62 | 16.21 | 24.88 | 27.16 | 32.29 | 30.0x |
| | PAT+GLAT+CTC | 25.12 | 26.99 | 22.22 | 26.52 | 30.58 | 32.27 | 20.5x |

Speedup. Speedup contains two indicators. Training indicates the training acceleration compared with the standard transformer model, and Inference indicates the acceleration of the inference process. Compared with AT models, the acceleration effects of different types of NAT models vary greatly. For iterative NAT models, inference speedup ranges from 1.7x to 4.0x. For the non-iterative NAT model, the inference acceleration can reach a significant 10.0x to 16.8x, which means that the non-iterative NAT model has a significant gap in inference speed compared to the iterative NAT model. PAT accelerates the inference speed to 15.5x, maintaining the inference acceleration effect of the general NAT models. PAT achieves 2.5x accelerated training of the NAT model, which means that under the same training settings, the training time is reduced by more than half.

Performance. The translation quality of all NAT models still lags behind the AT models. Compared with the non-iterative NAT model, PAT enables the NAT model to achieve the best performance in multiple directions, further narrowing the performance gap with AT models. It is worth noting that an open problem in the NAT model is the balance between accelerating decoding and maintaining performance. PAT can achieve training acceleration without affecting translation quality. This is the first time that a NAT model has achieved further acceleration without compromising performance. At the same time, PAT alleviates the disadvantage of the NAT model's acceleration imbalance in the training and inference phases. PAT achieves the best translation performance among all non-iterative NAT models.

3.6 Document-level translation results

As shown in Table 2, we verified the experimental results of PAT for document-level translation tasks. Document-level translation tasks are closer to real-life scenarios and more complex than sentence-level translation tasks. PAT enables the NAT model to be applied to both sentence-level and document-level translation tasks for the first time.

Speedup. Document-level translation is characterized by strong contextual relevance, so the input sequence is longer than the sentence-level translation task. Unlike sentence-level translation tasks, document-level translation tasks do not require a filling strategy, so speedup only represents decoding acceleration. Compared with the AT model, the decoding acceleration of the NAT model is more than 20x, which shows that the

decoding acceleration of the NAT model is more obvious in long sequence tasks. The Vanilla-NAT model has the most obvious acceleration effect, reaching 40.8x, and the PAT-GLAT decoding acceleration is 30.0x, which is 3.0x higher than GLAT+CTC. The inference acceleration effect of PAT+GLAT+CTC is 20.5x, which is slightly slower than other NAT models, but still significantly improves the inference speed compared with the AT model.

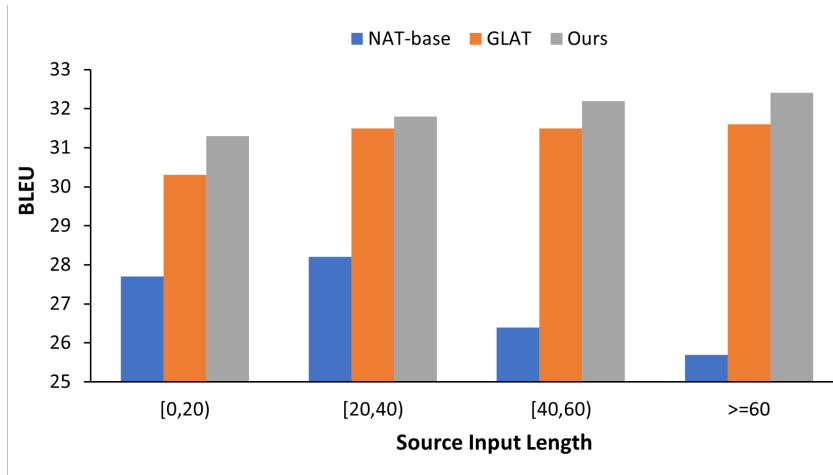


Fig. 3. Performance under different source input length on WMT14 EN \leftrightarrow DE.

Performance. The input sequence of document-level translation tasks is long, which brings challenges to the translation performance and stability of AT and NAT models. It is worth noting that for document-level translation tasks, PAT can be used in both AT and NAT models. In the AT model, the standard transformer model cannot produce effective results on the TED and News data sets, indicating that PAT can overcome the long sequence attention challenge and has better stability than the standard transformer. Among the experimental results obtained, the PAT+transformer model achieved the highest translation quality among all models. For the NAT model, PAT narrows the performance gap between the NAT model and the AT model. For example, PAT+GLAT+CTC achieves performance close to that of the AT model on the KD data of all data sets, while the AT model performs better than the KD data on the Raw data of each data set, which shows that knowledge distillation significantly improves the performance of NAT models.

3.7 Breakdown Results

As shown in Figure 3, we evaluated the impact of different input lengths on model performance. Our method performed significantly better than other models in short sentences, and scored the highest in sentence reasoning of different lengths, proving that our method has stable parallel decoding capabilities.

Table 3. Accuracy [%] of translation prediction for specific contextual phenomena (deixis, lexical consistency, ellipsis (inflection), and VP ellipsis) between different models on the English-Russian contrastive test set.

| Method | deixis | el.infl. | el.VP | lexcoh |
|-----------------|--------|----------|-------|--------|
| Transformer | 50.0 | 51.5 | 26.9 | 45.8 |
| PAT+Transformer | 78.6 | 80.1 | 71.5 | 52.6 |
| PAT+GLAT+CTC | 50.0 | 53.4 | 41.2 | 45.3 |

To determine if various models effectively use context to resolve discourse inconsistencies, we utilize contrastive test sets. These sets evaluate discourse phenomena in English-Russian translations, including deixis, lexicon consistency, ellipsis (inflection), and ellipsis (verb phrase). Each instance has a positive translation and several negative translations that differ by only one specific word. The objective is to assess whether a model is more likely to produce an accurate translation than incorrect variations. As shown in Table 3, The PAT+Transformer outperforms the NAT models across all four discourse phenomena. PAT+GLAT+CTC performs similarly to the Transformer baseline (without document context) on deixis and lexical cohesion (lexcoh), but outperforms it on the ellipsis of inflection (el.infl.) and ellipsis of verb phrase (el.VP).

4 Conclusion

In this paper, we propose Padding Accelerated Training (PAT) for NAT. In contrast to previous approaches, PAT does not use padding tokens but rather incorporates another real training sentence into the padding process. This method preserves the contextual information within the sentence and prevents interference between different sentences during training. The results demonstrated that PAT achieved the best performance in the single-pass decoding NAT models. Compared with the AT baseline model, PAT has improved the training speed by 2.5x, achieving the simultaneous acceleration of training and inference of the NAT model for the first time. Furthermore, we applied PAT to document-level translation tasks. The experimental results demonstrated that PAT enabled NAT to achieve multi-scenario translation tasks, thereby narrowing the performance gap between the NAT model and the AT model in document-level translation tasks.

References

1. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.
2. Gu J, Bradbury J, Xiong C, et al. Non-autoregressive neural machine translation[C]//International Conference on Learning Representations (ICLR). 2018.Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999)
3. Gu J, Wang C, Zhao J. Levenshtein transformer[J]. Advances in neural information processing systems, 2019, 32.

4. Ghazvininejad M, Levy O, Liu Y, et al. Mask-predict: Parallel decoding of conditional masked language models[J]. arxiv preprint arxiv:1904.09324, 2019.
5. Devlin J, Chang M W, Lee K, et al. Pre-training of deep bidirectional transformers for language understanding in: Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)[J]. Minneapolis, MN: Association for Computational Linguistics, 2019: 4171-86.
6. Ghazvininejad M, Karpukhin V, Zettlemoyer L, et al. Aligned cross entropy for non-autoregressive machine translation[C]//International Conference on Machine Learning. PMLR, 2020: 3515-3523.
7. Qian L, Zhou H, Bao Y, et al. Glancing transformer for non-autoregressive neural machine translation[J]. arxiv preprint arxiv:2008.07905, 2020.
8. Gu J, Kong X. Fully non-autoregressive neural machine translation: Tricks of the trade[J]. arxiv preprint arxiv:2012.15833, 2020.
9. Huang F, Zhou H, Liu Y, et al. Directed acyclic transformer for non-autoregressive machine translation[C]//International Conference on Machine Learning. PMLR, 2022: 9410-9428.
10. Hamming R W. Error detecting and error correcting codes[J]. The Bell system technical journal, 1950, 29(2): 147-160.
11. Papineni K, Roukos S, Ward T, et al. Bleu: a method for automatic evaluation of machine translation[C]//Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002: 311-318.
12. Liu Y, Gu J, Goyal N, et al. Multilingual denoising pre-training for neural machine translation[J]. Transactions of the Association for Computational Linguistics, 2020, 8: 726-742.
13. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network[J]. arxiv preprint arxiv:1503.02531, 2015.
14. Ott M, Edunov S, Baevski A, et al. fairseq: A fast, extensible toolkit for sequence modeling[J]. arxiv preprint arxiv:1904.01038, 2019.
15. Graves A, Fernández S, Gomez F, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks[C]//Proceedings of the 23rd international conference on Machine learning. 2006: 369-376.
16. Kasai J, Cross J, Ghazvininejad M, et al. Non-autoregressive machine translation with disentangled context transformer[C]//International conference on machine learning. PMLR, 2020: 5144-5155.
17. Ghazvininejad M, Levy O, Zettlemoyer L. Semi-autoregressive training improves mask-predict decoding[J]. arxiv preprint arxiv:2001.08785, 2020.
18. Huang X S, Perez F, Volkovs M. Improving non-autoregressive translation models without distillation[C]//International Conference on Learning Representations. 2021.
19. Libovický J, Helcl J. End-to-end non-autoregressive neural machine translation with connectionist temporal classification[J]. arxiv preprint arxiv:1811.04719, 2018.
20. Ran Q, Lin Y, Li P, et al. Guiding non-autoregressive neural machine translation decoding with reordering information[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2021, 35(15): 13727-13735.
21. Ma X, Zhou C, Li X, et al. Flowseq: Non-autoregressive conditional sequence generation with generative flow[J]. arxiv preprint arxiv:1909.02480, 2019.
22. Sun Z, Li Z, Wang H, et al. Fast structured decoding for sequence models[J]. Advances in Neural Information Processing Systems, 2019, 32.