INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

An Improved Single Step Non-autoregressive Transformer for Automatic Speech Recognition

Ruchao Fan[1], *Wei Chu*[2], *Peng Chang*[2], *Jing Xiao*[2], *Abeer Alwan*[1]

[1]Dept. of Electrical and Computer Engineering, University of California, Los Angeles, USA
[2]PAII Inc., USA

{fanruchao,alwan}@g.ucla.edu, {chuwei129,changpeng805, xiaojing661}@pingan.com.cn

Abstract

Non-autoregressive mechanisms can significantly decrease inference time for speech transformers, especially when the single step variant is applied. Previous work on CTC alignment-based single step non-autoregressive transformer (CASS-NAT) has shown a large real time factor (RTF) improvement over autoregressive transformers (AT). In this work, we propose several methods to improve the accuracy of the end-to-end CASS-NAT, followed by performance analyses. First, convolution augmented self-attention blocks are applied to both the encoder and decoder modules. Second, we propose to expand the trigger mask (acoustic boundary) for each token to increase the robustness of CTC alignments. In addition, iterated loss functions are used to enhance the gradient update of low-layer parameters. Without using an external language model, the WERs of the improved CASS-NAT, when using the three methods, are 3.1%/7.2% on Librispeech test clean/other sets and the CER is 5.4% on the Aishell1 test set, achieving a 7%$\sim$21% relative WER/CER improvement. For the analyses, we plot attention weight distributions in the decoders to visualize the relationships between token-level acoustic embeddings. When the acoustic embeddings are visualized, we find that they have a similar behavior to word embeddings, which explains why the improved CASS-NAT performs similarly to AT.

Index Terms: non-autoregressive transformer, CTC alignment, token-level acoustic embedding, end-to-end ASR

1. Introduction

Transformers have been dominant in many sequence generation tasks, outperforming their recurrent neural network (RNN) counterparts in terms of both accuracy and speed for end-to-end systems [1–3]. However, the autoregressive (left-to-right) generation order slows down inference speed significantly. To accelerate the inference, non-autoregressive transformers (NAT) were proposed for the parallel generation of the output sequence. The idea is widely adopted in neural machine translation (NMT) [4–6], automatic speech recognition (ASR) [7–18], text-to-speech (TTS) [19, 20] and speech translation [21].

Current NAT models for ASR can be categorized into: (i) iterative NAT, and (ii) single step NAT, according to the number of iterations for sequence generation. Essentially, autoregressive models are also iterative-based since they use a left-to-right generation order and take N iterations to generate a sequence of length N. Hence, the idea of iterative NAT is to adopt a different generation order with less than N iterations to accelerate the inference. Chen et al. regarded the transformer decoder as a masked language model that first generates tokens with high confidence [7], while Higuchi et al. applied the same idea but based on the connectionist temporal classification (CTC) output [11, 16]. In addition, Fujita et al. used the idea of the insertion transformer from NMT to generate the output sequence with an arbitrary order [12]. Another recent effective method is using multiple decoders as refiners to do an iterative refinement based on CTC alignments [14]. Theoretically, the iterative NAT has a limited improvement of inference speed since multiple iterations are still needed to obtain a competitive result. In contrast, single step NAT, which attempts to generate the output sequence with only one iteration, can have a better speed up for inference. The idea is to substitute the word embedding in autoregressive models with an acoustic representation for each output token, assuming that language semantics can also be captured by acoustic representations [9, 10, 13].

Although various NAT methods were proposed for ASR, the WER performance still lags behind that of state-of-the-art autoregressive models. Therefore, based on our previous work [13], we propose several methods to improve the accuracy of CTC alignment-based single step NAT (CASS-NAT) with little inference speed loss. First, convolution augmented self-attention blocks are applied to both the encoder and decoder modules, while other work only considered using them in the encoder [22]. The second method is to expand the trigger mask (acoustic boundary) for each token to increase the robustness of the CTC alignment. Third, considering the wide use of iterated loss functions to train deep transformers [23–25], we apply iterated loss to enhance the gradient update of low-layer parameters for both the encoder and decoder modules. When no external language model is used, large improvements are observed on both the Librispeech [26] and Aishell1 [27] datasets in terms of error rate, and the performance is close to the autoregressive baseline. Additionally, we analyse the self-attention distributions and token-level acoustic embeddings in the decoders. We find a similar behaviour between the token-level acoustic embedding and word embedding, which explains why CASS-NAT performs similarly to autoregressive models.

The remainder of the paper is organized as follows. Section 2 briefly reviews the CASS-NAT and describes the proposed methods for improving the system. Section 3 describes the recognition experimental setup, followed by results and analyses in Section 4. Section 5 concludes the paper.

2. System Overview

2.1. Basic CASS-NAT Model

The CTC alignment-based single step non-autoregressive transformer (CASS-NAT) that we proposed in [13] is modified based on the CTC/Attention hybrid architecture [28] to be non-autoregressive. Fig.1 shows four major modules in the CASS-NAT: encoder, token acoustic extractor (TAE), self-attention decoder (SAD) and mixed-attention decoder (MAD).

http://dx.doi.org/10.21437/Interspeech.2021-1955

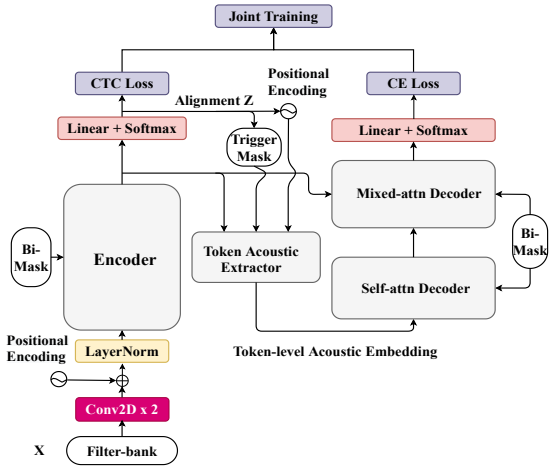


Figure 1: *An overview of the CASS-NAT architecture [13]*

Suppose the input sequence is $X = \{x_1, ..., x_t, ..., x_T\}$ and ground truth is $Y = \{y_1, ...y_u, ..., y_U\}$, the encoder extracts high level acoustic representations H from speech features X, followed by a CTC loss function. The role of CTC is to obtain an alignment over CTC output space to offer auxiliary information for the token acoustic extractor. Specifically, given an alignment $Z = \{z_1, ..., z_t, ..., z_T\}$, we can estimate an acoustic segment for each token u as $\{z_{t_{u-1}+1}, ...z_{t_u}\}$, as well as the number of tokens in Z. The segment is equivalent to the trigger mask in Fig. 1, which is used for self-attention computation. The information obtained from the CTC alignment is then used to extract token-level acoustic embeddings that replace word embeddings existing in autoregressive transformers (AT). The self-attention and mixed-attention decoders are finally used to model the relationship between tokens, where the MAD has access to the acoustic representations H, while the SAD does not.

The framework is trained by jointly optimizing a CTC loss function on the encoder side (L_{CTC}) and a cross entropy (CE) loss function on the decoder side (L_{dec}) so that the final loss function (L_{joint}) is defined as:

$$L_{\text{joint}} = \lambda \cdot \log \sum_{Z \in q} \prod_{i=1}^{T} P(z_i|X) + \log \prod_{u=1}^{U} P(y_u|z^*_{t_{u-1}+1:t_u}, X) \tag{1}$$

where P is the probability, λ is the task ratio of the CTC loss, and q is a set of alignments that can be mapped to the ground truth Y. Z^* is the most probable alignment, obtained by forced alignment over the CTC output space. Note that the second term is a maximum approximation for the loss on the decoder side.

2.2. Convolution Augmented Self-attention Block

Self-attention is the most basic layer in speech transformers, and is formulated as:

$$Attn(Q, K, V, M) = Softmax(\frac{QK^T}{\sqrt{d_k}}) \otimes M \cdot V \tag{2}$$

where $Q \in R^{n_q \times d_q}$, $K \in R^{n_k \times d_k}$, $V \in R^{n_v \times d_v}$, and $M \in R^{n_q \times n_k}$ are the queries, keys, values and mask matrix, respectively. From the equation, it can be seen that self-attention considers global information across the sequence, but ignores local details. To alleviate this problem, convolution augmented self-attention blocks are proposed to emphasise the modelling of local dependencies of the input sequence in the encoder [22,29]. Different from previous work, we apply the convolution augmented self-attention blocks in the mixed-attention decoder as well as the encoder. Specifically, the feed-forward layer is decomposed into two sub-layers to be placed at the beginning and the end of the block. A convolution layer similar to that in [22] is inserted after the self-attention layer except that we empirically use layer normalization instead of batch normalization. The final computation in the i^{th} MAD can be formulated as:

$$\hat{s_i} = s_i + \frac{1}{2}\text{FFN}(s_i) \tag{3}$$

$$s_i^{'} = \hat{s_i} + \text{LN}(\text{Attn}(\hat{s_i}, \hat{s_i}, \hat{s_i}, \text{BiMask})) \tag{4}$$

$$s_i^{''} = s_i^{'} + \text{Conv}(s_i^{'}) \tag{5}$$

$$s_i^{'''} = s_i^{''} + \text{LN}(\text{Attn}(s_i^{''}, H, H, \text{BiMask})) \tag{6}$$

$$o_i = \text{LN}(s_i^{'''} + \frac{1}{2}\text{FFN}(s_i^{'''})) \tag{7}$$

where LN indicates layer normalization and FFN is the feed-forward layer. BiMask stands for a bidirectional mask.

In addition, different from the usage of relative positional encoding in [22, 30], we consider a maximum length of relative position k as in [31]. Therefore, $2k+1$ position embedding are learned to represent the relative position between $[-k, k]$.

2.3. Trigger Mask Expansion

The quality of token-level acoustic embedding relies on the accuracy of the trigger mask, which is mapped from the CTC alignment. Although the CTC loss function is used to optimize the alignment, there are still errors when doing forced alignment over CTC output space, leading to an inaccurate trigger mask. In order to compensate the inaccuracy of token-level acoustic embedding extraction, we expand the trigger mask to include contextual frames for each token. For example, suppose the contextual frame size is one, the acoustic boundary of token U becomes $\{z_{t_{u-1}}, ..., z_{t_u+1}\}$. The trigger mask will then be expanded by one in the subsequent acoustic embedding extraction.

2.4. Iterated Loss

Deep transformers always suffer from gradient vanishing, especially for parameters that are distant from the output layers. Iterated loss are proposed to add additional loss functions after each layer to boost the gradient update [23, 32]. Recent work proposed iterated CTC loss to improve the performance of CTC [25]. In [24], iterated CE loss is used for training deep transformer-based acoustic models. Since CTC and CE loss functions are jointly optimized in the CASS-NAT framework, we integrate iterated CTC and CE loss functions into Eq.1 so that the parameters in different layers can be updated at the same scale. We find this strategy to be more effective for CASS-NAT than AT models. Let $L_{dec} = \log \prod_{u=1}^{U} P(y_u|z^*_{t_{u-1}+1:t_u}, X)$ and $L_{CTC} = \log \sum_{Z \in q} \prod_{i=1}^{T} P(z_i|X)$, the objective function is re-written as:

$$\begin{aligned} L_{\text{joint}} = {} & \lambda_{CE} L_{dec}^{final} + (1 - \lambda_{CE}) L_{dec}^{middle} \\ & + \lambda_{CTC} L_{CTC}^{final} + (1 - \lambda_{CTC}) L_{CTC}^{middle} \end{aligned} \tag{8}$$

where λ_{CE} and λ_{CTC} are task ratios. Middle and final indicate the layer position of the inserted loss functions.

Table 1: *WERs of the proposed methods for improving CASS-NAT on Librispeech. No external language model is used. SpecAug is used in all configurations. WERR is the incremental relative WER improvement on the test-other data.*

Model w/o LM	dev-clean	dev-other	test-clean	test-other	*WERR*
Conformer AT	2.7	7.2	3.0	7.0	
CASS-NAT	3.7	9.2	3.8	9.1	-
+ Conv-aug Enc.	3.1	7.9	3.3	7.9	13.2%
+ Conv-aug Dec.	3.0	7.8	3.1	7.6	3.8%
+ Tri. Mask Exp.	3.0	7.6	3.1	7.5	1.3%
+ Iterated CTC	2.8	7.3	3.1	7.3	2.7%
+ Iterated CE	2.8	7.3	3.1	7.2	1.4%

3. Experimental Setup

The experiments were conducted on three datasets: the 960-hour LibriSpeech English corpus [26], the 178-hour Aishell1 Mandarin corpus [27] and the 50-hour OGI Kids corpus (English) [33]. All experiments used 80-dim Mel-filter bank features, computed every 10ms with a 25ms Hamming window. Features of every 3 consecutive frames were concatenated to form a 240-dim feature vector as the input. The sets of output labels consist of 5k word pieces obtained by the SentencePiece method [34] for Librispeech and OGI. 4230 Chinese characters were obtained from the training set for the Aishell1 dataset.

A CTC/Attention AT baseline was first trained with the architecture ($N_e = 10$, $N_d = 5$, $d_{ff} = 2048$, $nh = 8$, $d_{att} = 512$) for Librispeech, ($N_e = 12$, $N_d = 6$, $d_{ff} = 2048$, $nh = 4$, $d_{att} = 256$) for the Aishell1 and ($N_e = 8$, $N_d = 4$, $d_{ff} = 2048$, $nh = 4$, $d_{att} = 256$) for the OGI. When training the CASS-NAT, the encoder was initialized with the AT encoder for faster convergence as in [35]. The decoder in the AT baseline was replaced by 1-block token-level acoustic extractor, 3-block self-attention decoder and 4-block mixed-attention decoder. The maximum length of relative position k was set to 20 in the encoder and 8 in the decoder for the English tasks and 4 in the decoder for the Aishell1 data. The contextual frame of the trigger mask expansion was 1. The iterated loss functions were inserted in the middle layer of the encoder and mixed-attention decoder with λ_{CE} of 0.9 and λ_{CTC} of 0.5. The inserted projection layers were not used during inference. These settings were empirically chosen based on many experiments.

Each of the two convolution layers in Fig.1 has 64 filters with a kernel size of 3 and a stride of 2, leading to a 4x frame-rate reduction. The same learning schedule in [13] was adopted. Layer normalization, dropout with rate of 0.1 and label smoothing with a penalty of 0.1 were all applied as the common strategies for training a transformer. We also applied SpecAug [36] for fair comparisons with results in previous literature. We additionally applied speed-perturbation for the Aishell1 dataset for fair comparisons with previously published results. We used development sets for early stopping and model averaging for final evaluation. Most of the experiments ended within 90 epochs. During AT decoding, the beam size is set to 20 for Librispeech, and it is set to 10 for the Aishel1 and OGI. The evaluation of the real time factor (RTF) was conducted using an NVIDIA Tesla V100 GPU with batch size of one.

A transformer-based language model was trained with the provided text in Librispeech for shallow fusion during AT baseline decoding. The language model was also used for ranking alignments and beam search during CASS-NAT decoding.

Table 2: *A comparison of error rates and RTFs with previously published results. RTFs of previous work are missing because the authors did not report them, or the machines used to test RTFs are different. †: use SpecAug. ⋆: use speed perturbation.*

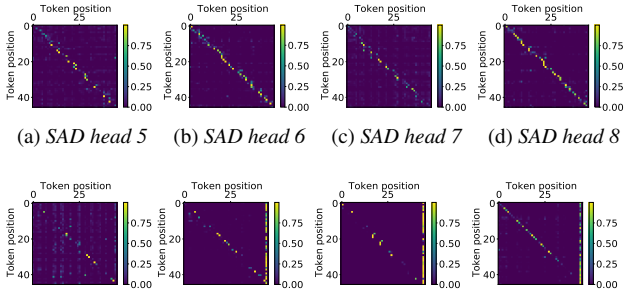
Librispeech (WER)	LM	test clean	test other	RTF test-other
Previous work (NAT)				
A-FMLM [7] †	w/o	6.6	12.2	-
Imputer [8]	w/o	4.0	11.1	-
Align-refine [14] †	w/o	3.6	9.0	-
CASS-NAT [13] †	w/o	3.8	9.1	0.010
Conformer AT†	w/o	3.0	7.0	0.499
	w/	2.3	5.2	0.568
Improved CASS-NAT †	w/o	**3.1**	**7.2**	0.014
	w/	2.8	6.5	0.188
Aishell1 (CER)	LM	dev	test	RTF test
Previous work (NAT)				
ST-NAT [10] †	w/o	6.9	7.7	-
A-FMLM [7] ⋆	w/o	6.2	6.7	-
Insertion-NAT [12] †	w/o	6.1	6.7	-
Enhanced-NAT [15] †⋆	w/o	5.3	5.9	-
BERT-LASO [18] †	w/o	5.2	5.8	-
CASS-NAT [13] †⋆	w/o	5.3	5.8	0.011
Conformer AT †⋆	w/o	4.8	5.2	0.200
Improved CASS-NAT †⋆	w/o	**4.9**	**5.4**	0.023

4. Results and Analyses

4.1. Results on Librispeech and Aishell1

Experiments are first conducted on the Librispeech corpus by incrementally adding the proposed methods based on the original CASS-NAT [13] as shown in Table 1. First, when convolution augmented self-attention blocks are applied to the encoder, the WER on the test-other set has a 13.2% relative improvement compared to the CASS-NAT baseline. When the convolution augmented blocks are also applied to the decoder, the WER drops further by a 3.8% relative improvement. An 1.3% relative WER improvement is observed for the trigger mask expansion method. When using the iterated loss function to both the encoder and decoder, we obtain an incremental 2.7% and 1.4% WER improvements, respectively. The final result has less than a 3% increase in relative WER compared to the AT baseline, which used a conformer structure. In the next sub-section, we will explain why the improved CASS-NAT can achieve a performance that is close to its autoregressive counterpart.

The proposed three methods are also used to train an improved CASS-NAT on the Aishell dataset. The final WER and real time factor (RTF) comparisons with previously published results are shown in Table 2, including both the Librispeech and Aishell1 data. Using the Librispeech dataset, the WER on test-other for the proposed methods has a 21% relative improvement over the original CASS-NAT with little RTF degradation (from 0.010 to 0.014). This RTF still has a 36x speed up compared to the AT baseline when no external LM is used. When using an external LM during inference, CASS-NAT does not benefit from LM as much as the AT baseline does, which was also reported in [13]. Using the Aishell1 dataset, the RTF speed up is not as large as in Lirbispeech. The reason may be that the speed up of NAT benefits longer utterances; however, there are more short utterances in Aishell1. In addition, we use AT baseline for ranking alignments, which may increase the computational cost. Although we do not carefully tune the hyper-parameters on Aishell1, the character error rate (CER) on the test set still improves from 5.8% to 5.4%, which is close to the AT baseline.



(a) *SAD head 5* (b) *SAD head 6* (c) *SAD head 7* (d) *SAD head 8*

(e) *MAD head 5* (f) *MAD head 6* (g) *MAD head 7* (h) *MAD head 8*

Figure 2: *Attention weight distributions of the last four heads in multi-head self-attention of the last block in the self-attention decoder (SAD) and mixed-attention decoder (MAD) for the first utterance in the Lirispeech train-clean-100 subset. The matrices Q, K, V in Eq.2 are split into nh sub-spaces and then they are used for self-attention computation separately. Each sub-space is referred to as a head. nh is set to 8 for this dataset.*

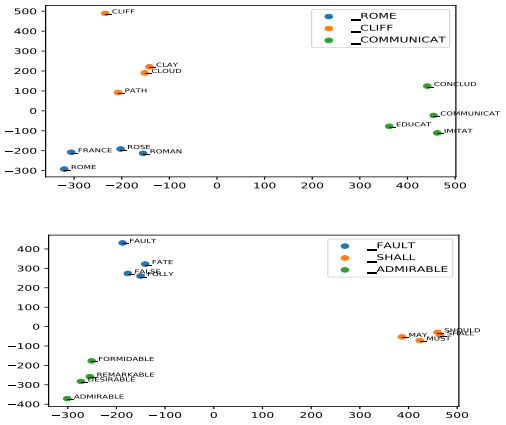


Figure 3: *Visualization of token-level acoustic embedding for six word pieces using the first two dimensions after PCA.*

4.2. Attention and Embedding Visualizations

To analyse the modelling of token-level acoustic embeddings, we choose the first utterance in the Librispeech train-clean-100 subset and plot the self-attention weights in the last block of the self-attention and mixed-attention decoders. The weights of the last 4 heads are shown in Fig.2. We can see from the figure that most of the heads learn a monotonic alignment between the token-level acoustic embeddings, indicating that each token relies more on adjacent tokens, which is similar to the idea of word embedding using continuous bag of words (CBOW) and skip-gram [37]. The monotonic alignment also shows the usefulness of the relative positional encoding because distant tokens with close semantic similarity have low attention weights.

To further investigate how the token-level acoustic embedding behaves, we extracted such embeddings from the first 3000 utterances in the train-clean-100 Librspeech subset. Each acoustic embedding has its own word piece ground truth. For each word piece, the embeddings are averaged to represent the final acoustic embedding. Using the same idea of visualizing word embedding, we randomly choose three word pieces and find their four closest embeddings using the cosine similarity distance. The 12 embeddings are reduced to a 2-dimensional space using principal component analyses (PCA) and are then plotted. Two examples are shown in Fig.3. The figure shows that the token-level acoustic embedding can learn not only the acoustic similarity, but also the word-piece level semantic similarity. For example, for the word piece **_ROME**, the closest tokens contain **_FRANCE** representing cities and countries, and **_ROSE** which has a similar pronunciation. The same situation can be seen in the lower part of the Fig.3 where each cluster has the same part-of-speech words, such as **_MAY** versus **_SHOULD** and **_REMARKABLE** versus **_FORMIDABLE**. This behaviour of token-level acoustic embedding is very similar to word embedding, indicating that they can also be used to capture semantics. This may explain why the improved CASS-NAT has a similar performance to the AT baseline.

4.3. NAT for Child Speech

Table 3: *WER for the development and test sets and RTF for the test set using the scripted part of the OGI corpus. Both experiments used SpecAug.*

	dev	test	RTF on test
Conformer AT	1.8	2.5	0.081
Improved CASS-NAT	2.2	2.6	0.018

In this section, we conduct experiments on the scripted part of the OGI kids corpus with the data partition in [38] except that 10% of the test set is chosen to be a development set for early stopping. The results are shown in Table 3. Since the OGI dataset has a fixed language pattern, the distributions of training and test sets are close. Although we trained the model within 15 epochs, the WER remains very competitive. Nevertheless, the conclusion of our improvements to CASS-NAT still holds and it has an impressive speed up than autoregressive models, which may be suitable for child ASR-based educational applications.

5. Summary and Conclusion

This paper presented three methods to improve the WER performance of CTC alignment-based single step NAT (CASS-NAT), followed by performance analyses. First, convolution augmented self-attention blocks were applied to the encoder and decoder modules. Second, the trigger mask was expanded for each token to compensate for the inaccuracy of CTC alignments. Third, iterated loss functions were used to enhance the gradient update of low-layer parameters. When using the three methods without external language models, we achieved a 7%$\sim$21% relative WER/CER improvement over the original CASS-NAT on the Librispeech and Aishell1 dataset with little RTF degradation. The WER performance was worse within 5%, in relative terms, than the autoregressive baseline, but maintained a 10$\sim$40x speed up. Moreover, attention weights and embedding visualization showed that the token-level acoustic embedding had similar behaviors with word embedding, explaining why the CASS-NAT performed similarly to AT.

6. Acknowledgements

This work was supported in part by the NSF.

7. References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6000–6010.

[2] J. Li, Y. Wu, Y. Gaur, C. Wang, R. Zhao, and S. Liu, "On the comparison of popular end-to-end models for large scale speech recognition," *Proc. Interspeech 2020*, pp. 1–5, 2020.

[3] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang *et al.*, "A comparative study on transformer vs rnn in speech applications," in *ASRU*. IEEE, 2019, pp. 449–456.

[4] J. Gu, J. Bradbury, C. Xiong, V. O. Li, and R. Socher, "Non-autoregressive neural machine translation," in *ICLR*, 2018.

[5] J. Lee, E. Mansimov, and K. Cho, "Deterministic non-autoregressive neural sequence modeling by iterative refinement," in *2018 EMNLP*. Association for Computational Linguistics, 2020, pp. 1173–1182.

[6] C. Saharia, W. Chan, S. Saxena, and M. Norouzi, "Non-autoregressive machine translation with latent alignments," in *EMNLP*, 2020, pp. 1098–1108.

[7] N. Chen, S. Watanabe, J. A. Villalba, P. Zelasko, and N. Dehak, "Non-autoregressive transformer for speech recognition," *IEEE Signal Processing Letters*, 2020.

[8] W. Chan, C. Saharia, G. Hinton, M. Norouzi, and N. Jaitly, "Imputer: Sequence modelling via imputation and dynamic programming," in *International Conference on Machine Learning*. PMLR, 2020, pp. 1403–1413.

[9] Y. Bai, J. Yi, J. Tao, Z. Tian, Z. Wen, and S. Zhang, "Listen attentively, and spell once: Whole sentence generation via a non-autoregressive architecture for low-latency speech recognition," *Proc. Interspeech 2020*, pp. 3381–3385, 2020.

[10] Z. Tian, J. Yi, J. Tao, Y. Bai, S. Zhang, and Z. Wen, "Spike-triggered non-autoregressive transformer for end-to-end speech recognition," *Proc. Interspeech 2020*, pp. 5026–5030, 2020.

[11] Y. Higuchi, S. Watanabe, N. Chen, T. Ogawa, and T. Kobayashi, "Mask ctc: Non-autoregressive end-to-end asr with ctc and mask predict," *Proc. Interspeech 2020*, pp. 3655–3659, 2020.

[12] Y. Fujita, S. Watanabe, M. Omachi, and X. Chang, "Insertion-based modeling for end-to-end automatic speech recognition," *Proc. Interspeech 2020*, pp. 3660–3664, 2020.

[13] R. Fan, W. Chu, P. Chang, and J. Xiao, "Cass-nat: Ctc alignment-based single step non-autoregressive transformer for speech recognition," in *ICASSP*. IEEE, 2021, pp. 5889–5893.

[14] E. A. Chi, J. Salazar, and K. Kirchhoff, "Align-refine: Non-autoregressive speech recognition via iterative realignment," *arXiv preprint arXiv:2010.14233*, 2020.

[15] X. Song, Z. Wu, Y. Huang, C. Weng, D. Su, and H. Meng, "Non-autoregressive transformer asr with ctc-enhanced decoder input," in *ICASSP*. IEEE, 2021, pp. 5894–5898.

[16] Y. Higuchi, H. Inaguma, S. Watanabe, T. Ogawa, and T. Kobayashi, "Improved mask-ctc for non-autoregressive end-to-end asr," in *ICASSP*. IEEE, 2021, pp. 8363–8367.

[17] Y. Fujita, S. Watanabe, and M. Omachi, "End-to-end asr and audio segmentation with non-autoregressive insertion-based model," *arXiv preprint arXiv:2012.10128*, 2020.

[18] Y. Bai, J. Yi, J. Tao, Z. Tian, Z. Wen, and S. Zhang, "Fast end-to-end speech recognition via non-autoregressive models and cross-modal knowledge transferring from bert," *arXiv preprint arXiv:2102.07594*, 2021.

[19] K. Peng, W. Ping, Z. Song, and K. Zhao, "Non-autoregressive neural text-to-speech," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7586–7598.

[20] C. Miao, S. Liang, M. Chen, J. Ma, S. Wang, and J. Xiao, "Flow-tts: A non-autoregressive network for text to speech based on flow," in *ICASSP*. IEEE, 2020, pp. 7209–7213.

[21] H. Inaguma, Y. Higuchi, K. Duh, T. Kawahara, and S. Watanabe, "Orthros: Non-autoregressive end-to-end speech translation with dual-decoder," in *ICASSP*. IEEE, 2021, pp. 7503–7507.

[22] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *Proc. Interspeech 2020*, pp. 5036–5040, 2020.

[23] A. Tjandra, C. Liu, F. Zhang, X. Zhang, Y. Wang, G. Synnaeve, S. Nakamura, and G. Zweig, "Deja-vu: Double feature presentation and iterated loss in deep transformer networks," in *ICASSP*. IEEE, 2020, pp. 6899–6903.

[24] Y. Wang, A. Mohamed, D. Le, C. Liu, A. Xiao, J. Mahadeokar, H. Huang, A. Tjandra, X. Zhang, F. Zhang *et al.*, "Transformer-based acoustic modeling for hybrid speech recognition," in *ICASSP*. IEEE, 2020, pp. 6874–6878.

[25] J. Lee and S. Watanabe, "Intermediate loss regularization for ctc-based speech recognition," in *ICASSP*. IEEE, 2021, pp. 6224–6228.

[26] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *ICASSP*. IEEE, 2015, pp. 5206–5210.

[27] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *20th O-COCOSDA*. IEEE, 2017, pp. 1–5.

[28] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[29] B. Yang, L. Wang, D. F. Wong, L. S. Chao, and Z. Tu, "Convolutional self-attention networks," 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)*, Jun. 2019, pp. 4040–4045.

[30] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2978–2988.

[31] P. Zhou, R. Fan, W. Chen, and J. Jia, "Improving generalization of transformer for speech recognition with parallel schedule sampling and relative positional embedding," *arXiv preprint arXiv:1911.00203*, 2019.

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[33] K. Shobaki, J.-P. Hosom, and R. A. Cole, "The ogi kids' speech corpus and recognizers," in *Sixth International Conference on Spoken Language Processing*, 2000.

[34] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *EMNLP 2018*, p. 66, 2018.

[35] R. Fan, P. Zhou, W. Chen, J. Jia, and G. Liu, "An online attention-based model for speech recognition," *Proc. Interspeech 2019*, pp. 4390–4394, 2019.

[36] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *Proc. Interspeech 2019*, pp. 2613–2617, 2019.

[37] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[38] R. Fan, A. Afshan, and A. Alwan, "Bi-apc: Bidirectional autoregressive predictive coding for unsupervised pre-training and its application to children's asr," in *ICASSP*. IEEE, 2021, pp. 7023–7027.