Interspeech 2024
1-5 September 2024, Kos, Greece

E-Paraformer: A Faster and Better Parallel Transformer for Non-autoregressive End-to-End Mandarin Speech Recognition

Kun Zou[1], *Fengyun Tan*[1], *Ziyang Zhuang*[1], *Chenfeng Miao*[1], *Tao Wei*[1], *Shaodan Zhai*[2], *Zijian Li*[3], *Wei Hu*[1], *Shaojun Wang*[1], *Jing Xiao*[1]

[1]Ping An Technology [2]Coupang [3]Georgia Institute of Technology

chinazoukun@gmail.com, joseewei@gmail.com, tanfengyun813@pingan.com.cn

Abstract

Paraformer is a powerful non-autoregressive (NAR) model for Mandarin speech recognition. It relies on Continuous Integrate-and-Fire (CIF) to implement parallel decoding. However, the CIF mechanism needs to recursively obtain the acoustic boundary of the emitted token, which will lead to inefficiency. In this paper, we introduce a novel monotonic alignment mechanism as an alternative to CIF that can convert frame-level embeddings into token-level embeddings in parallel. Combining this method with other improvements to the model structure, we design a faster and better parallel transformer called the Efficient Paraformer (E-Paraformer). Experiments are performed on the AISHELL-1 benchmark. Compared to Paraformer baseline, the E-Paraformer achieves character error rates (CER) of 4.36%/4.79% on the AISHELL-1 dev/test dataset, representing 7.8% and 6.3% (relative) reductions, respectively. Moreover, it achieves about 2x inference speedup and 1.35x training speedup.

Index Terms: ASR, E2E, non-autoregressive, single-step NAR, E-Paraformer

1. Introduction

Over the past few years, end-to-end (E2E) models have achieved remarkable results on automatic speech recognition (ASR) tasks. There are three popular E2E approaches: connectionist temporal classification (CTC) [1], recurrent neural network transducer (RNN-T) [2, 3], and attention-based encoder-decoder (AED) [4–6]. Among them, the RNN-T and AED are autoregressive (AR) models. ASR modeling has been dominated by AR models because of their superior accuracy. However, the AR decoder inside such AED models predicts the next token conditioned on all previous tokens step by step, which is computationally inefficient. Therefore, many non-autoregressive (NAR) methods are studied to achieve parallel decoding. The NAR model predicts tokens independently and simultaneously, bringing high inference speed and having very important engineering value, especially in offline ASR tasks.

A-FMLM [7] is the first attempt to introduce the conditional masked language model (CMLM) [8] into ASR modeling, proposing a non-autoregressive Transformer model (NAT). A-FMLM is designed to predict masked tokens conditioned on unmasked ones and whole speech embeddings. However, the length of the output tokens of the model needs to be predefined, which limits the performance of the model. To solve this problem, some methods use a CTC module to predict the length of the target sequence, such as ST-NAT [9]. Similarly, Mask-CTC and its variants [10–12] propose to use the CMLM decoder to refine CTC decodings. Imputer [13] provides a different way of NAR modeling through imputation and dynamic programming.

The NAR models introduced above are iterative. They require multiple iterations to achieve a competitive result, thus limiting the speed of inference in practice. To overcome this limitation, single-step NAR models are proposed. LASO [14] proposes a Position Dependent Summarizer (PDS) module to implement parallel decoding of Transformer style models, but it requires a predefined token length. InterCTC [15, 16] introduces an intermediate CTC loss to alleviate the conditional independence assumption of the CTC model, thus improving precision without compromising the inference speed. Several studies have conducted extensive investigations into the combination of CTC alignment and NAT, such as CASS-NAT and its various iterations [17–20]. These advancements have resulted in further enhancements in the performance of single-step NAR models.

In addition to the CTC-based or NAT-based NAR models mentioned above, Continuous Integrate-and-Fire (CIF) [21] parallelizes the decoder's calculations by directly predicting the length of the target sequence. It shows great promise in NAR modeling [22]. However, CIF is a soft and monotonic alignment mechanism, which has two disadvantages: 1) The acoustic boundary of each fired token is obtained recursively during inference and training, which will cause inefficiency. 2) Each fired token-level embeding is calculated based on a limited context. Paraformer [23] enhances the context modeling capabilities of CIF-based NAR models using an additional decoder forward pass and a Glancing Language Model (GLM)-based sampler. Although Paraformer achieves quite competitive performance on open source Mandarin Chinese datasets [24], it does not address the inefficient of CIF. The inference time of different Paraformer modules can be seen in Table 3.

Inspired by the Hard Monotonic Alignment (HMA) method proposed in [25], we introduce a monotonic attention matrix construction method as an alternative to CIF. It offers two advantages over CIF: 1) Parallel computation. It can directly convert frame-level acoustic embeddings into token-level acoustic embeddings in parallel. 2) Global context modeling. Each token-level acoustic embedding is calculated based on all frame-level acoustic embeddings. So we call it Parallel Integrate-and-Fire (PIF). We also present several supporting strategies to refine the performance of PIF-based model, such as introducing trainable hyperparameters and multi-head mechanism in PIF. In addition, some other modifications of model structure are made, such as removing the CTC loss and cross-attention layers in Paraformer to design a faster and better parallel transformer, which we call Efficient Paraformer (E-Paraformer). With the combined action of these methods, the proposed E-Paraformer achieves better CER performance than the Paraformer baseline in Mandarin speech recognition, as well as faster inference and training speeds.

10.21437/Interspeech.2024-1891

2. Related work

2.1. Continuous Integrate-and-Fire (CIF)

Continuous Integrate-and-Fire (CIF) is a soft and monotonic alignment mechanism employed in the E2E ASR model [21]. The calculation of CIF is illustrated in Fig. 1. CIF sequentially accumulates the weights $\boldsymbol{\alpha}$ obtained using a weight estimator and integrates frame-level acoustic embeddings $\boldsymbol{h}$. Once the accumulated weight reaches a given threshold β, the corresponding frame is located as the acoustic boundary and a token-level acoustic embedding $c_u (u = 1, 2, ...)$ is fired. It can be seen that the CIF process is recursive.

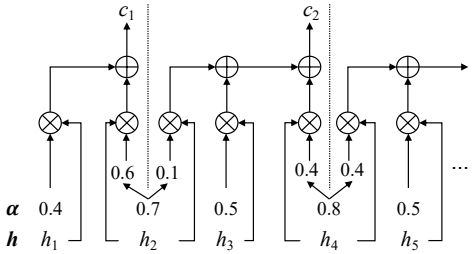


Figure 1: *Illustration of the CIF process (β is set to 1. $c_1 = 0.4 * h_1 + 0.6 * h_2$, $c_2 = 0.1 * h_2 + 0.5 * h_3 + 0.4 * h_4$.*

2.2. Paraformer and Sampler

Paraformer [23] is a CIF-based E2E NAR model, which mainly consists of conformer [26] encoder, parallel transformer [27] decoder, CIF-based predictor and GLM-based sampler. The sampler is used to improve the ability of the NAR decoder to model context interdependence.

Let the input sequence be $\boldsymbol{x} = (x_1, x_2, ..., x_T)$ and the output sequence be $\boldsymbol{y} = (y_1, y_2, ..., y_U)$, where T and U are the length of the acoustic frame sequence and the target transcription sequence, respectively. During training, Paraformer predicts the label $\boldsymbol{y}'$ conditioned on the semantic embedding $\boldsymbol{e}^{(\mathrm{s})}$ and the encoder output $\boldsymbol{h}$:

$$\boldsymbol{y}' = \mathrm{Decoder}(\boldsymbol{e}^{(\mathrm{s})}, \boldsymbol{h}) \tag{1}$$

where $\boldsymbol{h} = \mathrm{Encoder}(\boldsymbol{x})$ is the frame-level acoustic embedding, which is integrated into the decoder's computation via the cross-attention layer. $\boldsymbol{e}^{(\mathrm{s})}$ is computed by the sampler proposed in Paraformer, which is formulated as Eq. (2):

$$\boldsymbol{e}^{(\mathrm{s})} = \mathrm{Sampler}(\boldsymbol{c}, \boldsymbol{e}^{(\mathrm{t})}, N) \tag{2}$$

where $\boldsymbol{e}^{(\mathrm{t})} = \mathrm{Embed}(\boldsymbol{y})$ is the text embedding generated by the embedding layer. $\boldsymbol{c}$ represents the token-level acoustic embedding, which is calculated with the CIF-based predictor formulated as Eq. (3-4).

$$\boldsymbol{c} = \mathrm{CIF}(\boldsymbol{h}, \boldsymbol{\alpha}) \tag{3}$$

$$\boldsymbol{\alpha} = \mathrm{Sigmoid}(\mathrm{Linear}(\mathrm{Conv}(\boldsymbol{h}))) \tag{4}$$

The sampler incorporates text embedding $\boldsymbol{e}^{(\mathrm{t})}$ by randomly substituting N tokens into acoustic embedding $\boldsymbol{c}$ to generate semantic embedding $\boldsymbol{e}^{(\mathrm{s})}$. To calculate N, a first-pass decoding label $\boldsymbol{y}^*$ is calculated through an additional decoder forward process conditioned on $\boldsymbol{c}$ and $\boldsymbol{h}$, and then N is calculated based on the number of error tokens in $\boldsymbol{y}^*$. Which can be formulated as Eq. (5-6):

$$\boldsymbol{y}^* = \mathrm{Decoder}(\boldsymbol{c}, \boldsymbol{h}) \tag{5}$$

$$N = \left\lceil \gamma \sum_{u}^{U} (y_u \neq y_u^*) \right\rceil \tag{6}$$

where γ is a sampling factor to control the number of substituting tokens. It is set to 0.4 in this study.

During inference, the Paraformer decoder predicts tokens conditioned on $\boldsymbol{c}$ and $\boldsymbol{h}$ as shown in Eq. (5). To reduce the discrepancy between training and inference, an additional Cross-Entropy (CE) loss is used in the first-pass decoder [28]. We take the official open source version Paraformer in FunASR [1] [28] as our baseline. Its training objective includes the CTC loss, the final CE loss $\mathcal{L}_{\mathrm{CE}}(\boldsymbol{y}', \boldsymbol{y})$, quantity loss $\mathcal{L}_{\mathrm{QUA}}$ and the first-pass CE loss $\mathcal{L}_{\mathrm{CE}}(\boldsymbol{y}^*, \boldsymbol{y})$, which is formulated as Eq. (7):

$$\mathcal{L} = 0.3\mathcal{L}_{\mathrm{CTC}} + 0.7\mathcal{L}_{\mathrm{CE}}(\boldsymbol{y}', \boldsymbol{y}) + \mathcal{L}_{\mathrm{QUA}} + \mathcal{L}_{\mathrm{CE}}(\boldsymbol{y}^*, \boldsymbol{y}) \tag{7}$$

where the quantity loss $\mathcal{L}_{\mathrm{QUA}}$ is formulated as:

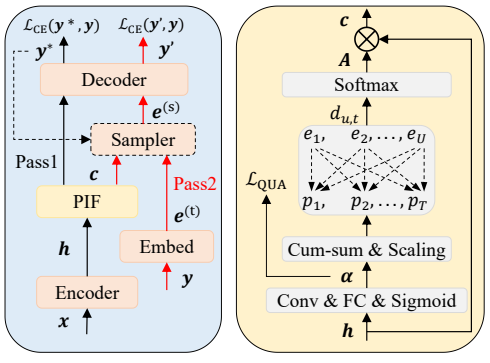
$$\mathcal{L}_{\mathrm{QUA}} = \left| \sum_{t=1}^{T} \alpha_t - U \right| \tag{8}$$

3. Proposed method

In this section, we will first introduce the proposed monotonic alignment mechanism called Parallel Integration-and-Fire (PIF), which serves as the foundation for the development of the new NAR model E-Paraformer. Then we will give a detailed introduction to the structural design of E-Paraformer.

3.1. Parallel Integrate-and-Fire (PIF)

Inspired by the Hard Monotonic Alignment (HMA) method applied in [25, 29, 30], we propose a novel monotonic alignment mechanism called Parallel Integrate-and-Fire (PIF). Compared with CIF, it has two advantages: parallel computing and global context modeling. The schematic diagram of the PIF calculation is presented in Fig. 2(b).



(a) E-Paraformer (b) PIF

Figure 2: *The model architecture of the proposed E-Paraformer and PIF module. (Operation* Cum-sum *is defined as Eq. (9) and Operation* Scaling *is defined as Eq. (10).)*

As in CIF-based models, PIF first takes the encoded outputs $\boldsymbol{h}$ to predict the acoustic information weights $\boldsymbol{\alpha}$ using the weight estimator that formulated as Eq. (4) and uses the quantity loss to predict the length of the target tokens. Then, PIF

[1] https://github.com/alibaba-damo-academy/FunASR.git

aims to integrate the frame-level acoustic embedding $\boldsymbol{h}$ in parallel. The key is to construct an alignment matrix based on α, and the specific steps can be expressed as Eq. (9-11).

First, a monotonically increasing alignment position vector $\boldsymbol{p}'$ is generated by accumulating α_t $(t = 1, 2, ...T)$, as shown in Eq. (9). Then, a scaling strategy shown in Eq. (10) is applied to ensure that the maximum value of $\boldsymbol{p}'$ is equal to the length of the target tokens.

$$p_t{}' = \sum_{m=1}^{t} \alpha_m, 1 \leq t \leq T \tag{9}$$

$$p_t = p_t{}' * \frac{U}{\sum_{i=1}^{T} \alpha_i}, 1 \leq t \leq T \tag{10}$$

According to the CIF mechanism, for a sequence of U tokens, its acoustic boundary vector is $\boldsymbol{b} = [0, 1, ..., U]$ $(\beta = 1)$. We define the center of each two adjacent acoustic boundaries as the acoustic center and denote it as $\boldsymbol{e} = [0.5, 1.5, ..., U - 0.5]$. The alignment position vector $\boldsymbol{p}$ now incorporates information about the acoustic boundaries. The left and right acoustic boundaries of calculating token-level embedding c_u are $u-1$ and u respectively. In CIF, c_u can be computed approximately as $c_u = \sum_t \{\alpha_t h_t | u - 1 \leq p_t \leq u\}$. While in PIF, c_u is computed by considering all frame-level embeddings, which is called global context modeling. We assume that frame-level acoustic embedding h_i contributes the most to generating c_u. To determine h_i, we find the embedding h_t closest to the center $u - 0.5$ based on its alignment position p_t, which can be expressed as $h_i = \{h_t | \mathrm{argmin}_t || p_t - (u - 0.5)| |_2,\ 1 \leq t \leq T\}$. Therefore, the construction method for the alignment matrix is based on the distances between the acoustic center $\boldsymbol{e}$ and the alignment position $\boldsymbol{p}$, as shown in Eq. (11):

$$A_{u,t} = \frac{\exp(-d_{u,t} * \sigma^{-2} + \delta)}{\sum_{t=1}^{T} \exp(-d_{u,t} * \sigma^{-2} + \delta)} \tag{11}$$

where $d_{u,t} = (e_u - p_t)^2, 1 \leq u \leq U, 1 \leq t \leq T$, which can be understood as the semantic distance between the u-th token and t-th encoder frame. $e_u = u - 0.5$ is the acoustic center emitting the u-th token. Softmax normalization is applied to ensure $\sum_{t=1}^{T} A_{u,t} = 1$. Obviously, the smaller semantic distance $d_{u,t}$ is, the larger attention weight $A_{u,t}$. The token-level embedding c_u is calculated as Eq. (12).

$$c_u = \sum_{t=1}^{T} A_{u,t} h_t \tag{12}$$

Trainable $\boldsymbol{\sigma}$&$\boldsymbol{\delta}$ strategy. Unlike the HMA method proposed in [25], we introduce two trainable hyperparameters σ and δ in Eq. (11) to improve alignment modeling capabilities. It significantly improves the performance of the model. Please see Table 2 for details.

Multi-head strategy. We use multiple σ and δ to build multi-head alignment matrix $\boldsymbol{A}^{(\mathrm{MH})}$ as follows:

$$\boldsymbol{A}^{(\mathrm{MH})} = \mathrm{Concat}(\boldsymbol{A}^1, \boldsymbol{A}^2, ..., \boldsymbol{A}^M) \tag{13}$$

$$\boldsymbol{A}^m = \frac{\exp(-d_{u,t} * \sigma_m^{-2} + \delta_m)}{\sum_{t=0}^{T} \exp(-d_{u,t} * \sigma_m^{-2} + \delta_m)} \tag{14}$$

where M is the number of heads.

Padding strategy. We also introduce a tag <sos/eos> at the beginning and end of the target sequence to teach the model to predict the beginning and end of the sentence, which proved to be beneficial to the performance of the model.

3.2. E-Paraformer

Based on the proposed PIF method, we design a new NAR architecture with faster training and inference speeds, which we call the Efficient Parallel Transformer (E-Paraformer). As shown in Fig. 2(a), E-Paraformer consists of the encoder, the PIF module, the decoder and the sampler. Its structure is similar to that of Paraformer, and the main improvement is to use the PIF module to replace the Paraformer CIF module. In addition to that, our decoder is built with the Transformer [27] encoder block, which excludes the cross-attention layer, while the Paraformer decoder is built with the Transformer decoder block, which includes the cross-attention layer. It can further speed up the decoder calculations. Our encoder is built with Conformer [26] blocks, which is same as Paraformer. We also use the sampler to enhance the semantic modeling capabilities of the model. Therefore, during training, the decoder has two forward processes: Pass1 and Pass2, which are illustrated by different color arrows in Fig. 2(a). Pass1 and Pass2 decoder process can be summarized as $\boldsymbol{y}^* = \mathrm{Decoder}(\boldsymbol{c})$ and $\boldsymbol{y}' = \mathrm{Decoder}(\mathrm{e}^{(\mathrm{s})})$ respectively. During inference, Pass1 process is performed.

The overall training objective of E-Paraformer is a combination of CE losses and quantity loss:

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}(\mathrm{y}',\mathrm{y})} + \lambda_1 \mathcal{L}_{\mathrm{QUA}} + \lambda_2 \mathcal{L}_{\mathrm{CE}(\mathrm{y}^*,\mathrm{y})} \tag{15}$$

where λ_1 and λ_2 are set to 1.0 in this study.

4. Experiments

4.1. Experimental setup

The proposed method is evaluated on AISHELL-1 [24], an open source 178-hour Mandarin speech corpus. The 80-channel filter-banks are used as features, calculated based on a 25 ms window with a stride of 10 ms. SpecAugment [31] and speed perturbation are used for data augmentation for all experiments. The layers of {encoder, decoder} are set to {12, 6}. The attention head is 4, with hidden dimension 256. The trainable hyperparameters σ and δ in Eq. (11) are initialized with 0.5 and 0.0 respectively. All our models are developed using the E2E speech recognition toolkit FunASR [1] [28]. Detailed information on other common training configurations, including optimizer settings, learning rate, etc., can be found in that publication. All experiments are performed on 8 NVIDIA Tesla V100 GPUs. We will open source our approach soon.

4.2. Results on AISHELL-1 task and analysis

In Table 1, we compare the proposed E-Paraformer with recently published NAR models in terms of CER and RTF. It can be seen that the CER performance of our model outperforms that of all other NAR models. Specifically, the proposed E-Paraformer achieves a CER of $4.36\%/4.79\%$ on the AISHELL-1 dev/test datasets. Even compared to the state-of-the-art (SOTA) Mandarin speech recognition model Paraformer, our model achieves an absolute reduction in CER of 0.24% and 0.41%, respectively. As we best known, it achieves a SOTA CER performance among NAR models on AISHELL-1 task. Furthermore, the proposed model achieves more than 2x inference speedup over Paraformer, with an RTF of 0.0069. The CER and RTF values of the previous works in Table 1 are from their papers. As we all know, differences in test devices such as GPU or CPU have a great impact on RTF calculations. Therefore, we make a more fair and detailed comparison in Table 2.

Table 1: *CER(%) and RTF results of the proposed E-Paraformer on AISHELL-1 tasks and comparison with previous leading NAR models. (†: RTF is evaluated with batch size of 8.)*

Model	dev/test	Params	RTF (↓)
A-FMLM [7]	6.2/6.7	-	0.2800
CTC-enhanced [12]	5.3/5.9	29.7M	0.0037 †
LASO-big [14]	5.9/6.6	80.0M	0.0040
Mask-CTC [10, 16]	5.2/5.7	-	0.0420
Improved CASS-NAT [18]	4.9/5.4	38.3M	0.0230
AL-NAT [19]	4.9/5.3	71.3M	0.0050
Paraformer [23]	4.6/5.2	46.2M	0.0168
E-Paraformer	**4.36/4.79**	43.6M	0.0069

In Table 2, we describe how to build the proposed E-Paraformer step by step based on Paraformer and give the CER and RTF results for each model. First, we reproduce the Paraformer baseline and perform ablation experiments on its sampler module and CTC loss. It can be seen that the sampler improves CER performance significantly, while the improvement brought by CTC is limited. We proceed to construct the S3 model by replacing the CIF module of the S2 model with a simplified version of the PIF module that does not include all supporting strategies: Trainable σ&δ, Multi-head and Padding. Now, the CER results of the S3 model are very close to those of Paraformer, and benefiting from the parallel computing of PIF, its inference speed is significantly improved. From the S3 model to the S5 model, we apply the **Trainable $\boldsymbol{\sigma}$&$\boldsymbol{\delta}$ strategy** and the **Padding strategy** respectively. The results show that the **Trainable $\boldsymbol{\sigma}$&$\boldsymbol{\delta}$ strategy** significantly improves the CER performance of the model, and the **Padding strategy** also has certain benefits for the model's CER performance. Because PIF already has global context modeling capabilities, we consider that the cross-attention layer in the Paraformer decoder has limited improvement in CER performance, so we remove the cross-attention layer (srcATT) in the decoder to build the S6 model. Based on the experimental results, upon removing the cross-attention layer, a marginal loss in CER is observed while significantly accelerating the inference speed. After that, we apply the 4-head **Multi-head strategy** on the S6 model, which further improves the CER performance of the model. Finally, we enhance the S7 model using the sampler module to obtain the proposed E-Paraformer model. The CER and RTF performance of E-Paraformer reaches the best level among all models.

Table 2: *Step-by-step process of building the proposed E-Paraformer based on Paraformer baseline. All models are trained and inferred on the same device. The RTF is calculated on the AISHELL-1 test set with a batch size of* 1*. The smaller the RTF, the faster the inference speed.*

Models	dev	test	RTF(↓)
S0: **Paraformer**	4.73	5.11	0.0136
S1: S0 - Sampler (**CIF**)	4.82	5.31	0.0135
S2: S1 - CTC	4.89	5.37	0.0136
S3: S2 + PIF	4.72	5.19	0.0076
S4: S3 + Trainable σ&δ	4.57	4.97	0.0075
S5: S4 + Padding	4.50	4.95	0.0076
S6: S5 - srcATT	4.52	4.97	0.0069
S7: S6 + Multi-head	4.46	4.92	0.0069
S8: S7 + Sampler (**E-Paraformer**)	**4.36**	**4.79**	**0.0069**

In order to more intuitively present the training and inference efficiency of different models. We select some critical node models to visualize their training and inference times. The results are shown in Tabel 3. The inference time of the predictor is significantly reduced from S0 to S3, mainly due to the effect of PIF. Because removing the sampler and CTC loss is only related to training. In addition, the training speed is increased by about 36%. From S3 to S8, the reduction in inference time is mainly reflected in the decoder while the inference time of other modules is almost the same. This is the effect of removing the cross-attention layer of the decoder. At the same time, although the S8 model adds a sampler, its training efficiency is almost unaffected due to the removal of the cross-attention layer in the decoder.

Table 3: *Comparison of training and inference efficiency of some key node models. The inference time is calculated on the entire AISHELL-1 test set. Predictor represents CIF module in S0 while represents PIF module in S3 and S8.*

Models	Inference time(s) (Encoder/Predictor/Decoder)	Training time (min/epoch)
S0	151.2/190.8/55.6	42.1(1.0x)
S3	150.6/8.0/55.4	30.9(1.36x)
S8	149.7/8.0/32.9	31.1(1.35x)

To further analyze the behaviour of the proposed PIF method, we plot the alignment matrices generated by PIF in Fig. 3. Here, we only show two of all four alignment matrices. As can be seen, both alignment plots are monotonic and clear, but they learn different contextual representations, with the second alignment capturing a broader context.

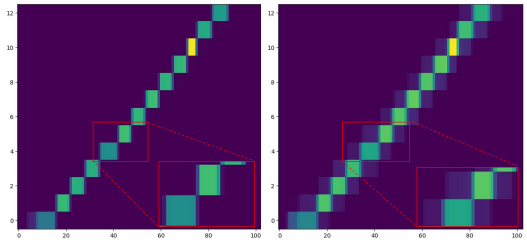


Figure 3: *Alignment plots generated by the proposed PIF method. We select two of total four attention heads for presentation. The horizontal axis represents the input frame step, and the vertical axis represents the output step.*

5. Conclusion

CIF-based NAR models like Paraformer have achieved leading results in Mandarin speech recognition. However, the recursive mechanism in CIF reduces inference efficiency. To overcome this, we propose a novel monotonic alignment mechanism called PIF. PIF not only enables parallel computation but also incorporates global context modeling capabilities. Using PIF, we design the improved NAR model E-Paraformer. Experiments on AISHELL-1 demonstrate that E-Paraformer surpasses the Paraformer baseline in CER performance and achieves faster training and inference speeds. PIF holds potential for application in other sequence-to-sequence tasks. In the future, we aim to explore the application of E-Paraformer in English speech recognition and the use of PIF in text-speech cross-modal representation learning.

6. References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[2] A. Graves, "Sequence transduction with recurrent neural networks," in *ICML*, 2012.

[3] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*, 2013.

[4] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *NIPS*, 2015.

[5] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *ICASSP*, 2016.

[6] C. Miao, K. Zou, Z. Zhuang, T. Wei, J. Ma, S. Wang, and J. Xiao, "Towards efficiently learning monotonic alignments for attention-based end-to-end speech recognition." *INTERSPEECH*, 2022.

[7] N. Chen, S. Watanabe, J. Villalba, P. Żelasko, and N. Dehak, "Non-autoregressive transformer for speech recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 121–125, 2020.

[8] M. Ghazvininejad, O. Levy, Y. Liu, and L. Zettlemoyer, "Mask-predict: Parallel decoding of conditional masked language models," in *EMNLP-IJCNLP*, 2019, pp. 6112–6121.

[9] Z. Tian, J. Yi, J. Tao, Y. Bai, S. Zhang, and Z. Wen, "Spike-triggered non-autoregressive transformer for end-to-end speech recognition," *INTERSPEECH*, 2020.

[10] Y. Higuchi, S. Watanabe, N. Chen, T. Ogawa, and T. Kobayashi, "Mask CTC: Non-autoregressive end-to-end ASR with CTC and mask predict," *INTERSPEECH*, 2020.

[11] Y. Higuchi, H. Inaguma, S. Watanabe, T. Ogawa, and T. Kobayashi, "Improved mask-ctc for non-autoregressive end-to-end asr," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 8363–8367.

[12] X. Song, Z. Wu, Y. Huang, C. Weng, D. Su, and H. Meng, "Non-autoregressive transformer asr with ctc-enhanced decoder input," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5894–5898.

[13] 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.

[14] 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," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1897–1911, 2021.

[15] J. Lee and S. Watanabe, "Intermediate loss regularization for ctc-based speech recognition," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6224–6228.

[16] J. Nozaki and T. Komatsu, "Relaxing the conditional independence assumption of CTC-based ASR by conditioning on intermediate predictions," *INTERSPEECH*, 2021.

[17] R. Fan, W. Chu, P. Chang, and J. Xiao, "CASS-NAT: CTC alignment-based single step non-autoregressive transformer for speech recognition," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5889–5893.

[18] R. Fan, W. Chu, P. Chang, J. Xiao, and A. Alwan, "An improved single step non-autoregressive transformer for automatic speech recognition," *INTERSPEECH*, 2021.

[19] Y. Wang, R. Liu, F. Bao, H. Zhang, and G. Gao, "Alignment-learning based single-step decoding for accurate and fast non-autoregressive speech recognition," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8292–8296.

[20] R. Fan, W. Chu, P. Chang, and A. Alwan, "A CTC Alignment-Based Non-Autoregressive Transformer for End-to-End Automatic Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 1436–1448, 2023.

[21] L. Dong and B. Xu, "CIF: Continuous integrate-and-fire for end-to-end speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6079–6083.

[22] F. Yu, H. Luo, P. Guo, Y. Liang, Z. Yao, L. Xie, Y. Gao, L. Hou, and S. Zhang, "Boundary and context aware training for cif-based non-autoregressive end-to-end asr," in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2021, pp. 328–334.

[23] Z. Gao, S. Zhang, I. McLoughlin, and Z. Yan, "Paraformer: Fast and accurate parallel transformer for non-autoregressive end-to-end speech recognition," *INTERSPEECH*, 2022.

[24] 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 *O-COCOSDA*, 2017.

[25] C. Miao, L. Shuang, Z. Liu, C. Minchuan, J. Ma, S. Wang, and J. Xiao, "EfficientTTS: An efficient and high-quality text-to-speech architecture," in *ICML*, 2021.

[26] 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," in *INTERSPEECH*, 2020.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *NIPS*, 2017.

[28] Z. Gao, Z. Li, J. Wang, H. Luo, X. Shi, M. Chen, Y. Li, L. Zuo, Z. Du, Z. Xiao *et al.*, "Funasr: A fundamental end-to-end speech recognition toolkit," *INTERSPEECH*, 2023.

[29] Z. Zhuang, K. Zou, C. Miao, M. Fang, T. Wei, Z. Li, W. Hu, S. Wang, and J. Xiao, "Improving attention-based end-to-end speech recognition by monotonic alignment attention matrix reconstruction," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 10 546–10 550.

[30] C. Miao, Q. Zhu, M. Chen, J. Ma, S. Wang, and J. Xiao, "Efficienttts 2: Variational end-to-end text-to-speech synthesis and voice conversion," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pp. 1650–1661, 2024.

[31] 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," in *INTERSPEECH*, 2019.