ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-6654-0540-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICASSP43922.2022.9746227

ALIGNMENT-LEARNING BASED SINGLE-STEP DECODING FOR ACCURATE AND FAST NON-AUTOREGRESSIVE SPEECH RECOGNITION

Yonghe Wang, Rui Liu[*]*, Feilong Bao, Hui Zhang, Guanglai Gao*

College of Computer Science, Inner Mongolia University, Hohhot, China
Inner Mongolia Key Laboratory of Mongolian Information Processing Technology, Hohhot, China
National & Local Joint Engineering Research Center of Intelligent Information Processing Technology for Mongolian, Hohhot, China

{cswyh92,liurui_imu}@163.com, {csfeilong,cszh,csggl}@imu.edu.cn

ABSTRACT

Non-autoregressive transformer (NAT) based speech recognition models have gained more and more attention since they perform faster inference speed compared with autoregressive counterparts, especially when the single-step decoding is applied. However, the single-step decoding process with length prediction will suffer from the decoding stability problem and limited improvement for inference speed. To address this, in this paper, we propose an alignment learning based NAT model, named AL-NAT. Our idea is inspired by the fact that the encoder CTC output and the target sequence are monotonically related. Specifically, we design an alignment cost matrix between the CTC output tokens and the target tokens and define a novel alignment loss to minimize the distance between the alignment cost matrix and the ground truth monotonic alignment path. By eliminating the length prediction mechanism, our AL-NAT model achieves remarkable improvements in recognition accuracy and decoding speed. To learn the contextual knowledge to improve the decoding accuracy, we further add lightweight language model on both the encoder and decoder side. Our proposed method achieves WERs of 2.8%/6.3% and RTF of 0.011 on Librispeech test clean/other sets with a lightweight 3-gram LM, and a CER of 5.3% and RTF of 0.005 on Aishell1 without LM, respectively.

Index Terms— Non-autoregressive transformer, Speech recognition, Alignment learning

1. INTRODUCTION

In recent years, end-to-end (E2E) automatic speech recognition (ASR) approaches have gained more and more attention in speech recognition community [1, 2]. The attention-based encoder-decoder models use an autoregressive modeling method to achieve the state-of-the-art ASR performance [3, 4], especially the successful application of transformer model [5–7]. However, during inference, such autoregressive models follow the probabilistic chain rule to predict the target sequence (with L tokens) from left to right. Note that such inference process adopts a time-consuming L-step iterative decoding process, that suffers from the slow inference speed. To this end, multi-step non-autoregressive transformer (NAT) models are proposed [8–10]. Specifically, such approaches predict the target sequence in a certain priority rather than in order. However, multi-step NAT methods provide limited improvement in inference speed because of the iterative decoding process.

Recently, single-step based NAT models [11–13] are favored for its fast decoding process. The key idea of single-step NAT is to substitute the decoder input word embedding in autoregressive models with an acoustic representation for each encoder output token [14–17]. During model training, the cross-entropy (CE) loss function is used on the decoder side. Specifically, Tian et al. [14] proposed the spike-triggered NAT model, which uses the connectionist temporal classification (CTC) spike output of the encoder as decoder input to predict the length of the target sequence. Bai et al. [18, 19] adopted a stack of attention blocks to reorganize the encoder representation to extract the token-level semantic, where the number of extracted token length is treated as the target sequence length. Fan et al. [16, 17] proposed to use the CTC alignment information of the encoder to extract the token-level acoustic embedding and predict the target sequence length.

However, such single-step NAT models with length prediction will suffer from two problems. The first one is the decoding instability problem. Inaccurate length prediction can lead to some unpredictable errors in the decoding sequence. Specifically, in [14], the available number of the CTC spikes is determined by an adjustable threshold, so it is easy to have a length mismatch. In [18] and [19], extracted token length is often much longer than the actual length of the target sequence due to the attention collapse. The second one is the limited inference speed. For example, the redundant symbols and blanks of the CTC output are still not completely removed [16, 17], thus affects the decoding speed.

For speech recognition, the input acoustic sequence and output tokens clearly perform a monotonic relationship [2]. CTC can be used as an auxiliary task for the AT model [2] to effectively calculate the optimal alignment between the input speech frames and the output tokens using dynamic programming algorithm. Therefore, there exists a natural monotonic alignment relationship between the CTC output and the target sequence.

Inspired by this, in this paper, we propose a novel alignment learning based NAT model, named AL-NAT, for accurate and fast token prediction. We propose two novel mechanisms: 1) we design

This research is supported by the National Key Research and Development Program of China (No.2018YFE0122900), China National Natural Science Foundation (No.61866030, No.62066033), Inner Mongolia Natural Science Foundation (No.2018MS06006), Applied Technology Research and Development Program of Inner Mongolia Autonomous Region (No.2019GG372, No.2020GG0046, No.2021GG0158, No.2020PT0002), Achievement Transformation Program of Inner Mongolia Autonomous Region (No.2019CG028) and Higher Educational Scientific Research Program of Inner Mongolia Autonomous Region (No.NJZY19011).

an alignment cost matrix between the CTC output tokens and the target tokens, to summarize the alignment information between them; 2) we define a novel alignment loss to minimize the distance between the alignment cost matrix and the ground truth monotonic alignment path. In such a way, our proposed AL-NAT contributes to predicting the target sequence directly without length prediction. Note that the CTC output tokens in our work do not include any blank and repetition symbols, etc., resulting in a fast inference speed. Furthermore, considering that the accuracy of encoder output will directly affect the performance of the final system and the NAT model cannot model the temporal context dependencies between the output. We propose to introduce a lightweight language model (LM) for both encoder and decoder to improve the inter-dependencies between the outputs labels.

We conduct our experiments on LibriSpeech [20] and Aishell1 [21] datasets. Experimental results show that AL-NAT achieves lower word/character error rates with fast inference compared with all baseline systems.

2. AL-NAT: METHODOLOGY

We propose a novel alignment learning based NAT model, denoted as AL-NAT, to achieve a fast and accurate ASR system. In this section, we first describe the model overview of AL-NAT, then explain the alignment learning loss in detail. Lastly, we describe how the newly added language model works during model inference.

2.1. Model overview

The overall architecture of the proposed AL-NAT is shown in Fig. 1, which consists of an encoder, a CTC module, an additional lightweight language model and a decoder.

Given the acoustic feature sequence $\boldsymbol{x} = \{x_1, x_2, ..., x_T\}$ with length T and the transcription sequence $\boldsymbol{y} = \{y_1, y_2, ..., y_L\}$ with length L. The encoder can be regarded as a feature extractor, which converts the input acoustic feature sequence $\boldsymbol{x}$ to a high-level representation $\mathcal{H}$.

The encoder is built based on conformer model [22]. We put two front-end convolution (Conv) blocks before the conformer blocks. Each conformer block is composed of four submodules: a multihead attention (MHA) module [23], a Conv module and a pair of position feed forward (FNN) modules are used at the front and rear. Each submodule is wrapped by residual connections and layer normalization. Among them, the Conv module can efficiently capture the local related information synchronously.

After that, the CTC module aims to learn the mapping between high-level acoustic representations $\mathcal{H}$ and transcription sequence $\boldsymbol{y}$ of different lengths and generates multiple approximately correct results that are represented by multiple CTC paths $\boldsymbol{c}$.

The CTC path $\hat{c}$ that can be obtained using the greedy algorithm:

$$\hat{c} = \arg\max_{c} \prod_{t=1}^{T} P(c_t|\mathbf{x}; \theta_{enc}) \tag{1}$$

where θ_{enc} indicates the parameters of the encoder. We remove all meaningless symbols in the CTC path $\hat{c}$ to obtain a compact token sequence $\boldsymbol{c} = \{c_1, c_2, ..., c_K\}$ (K means the length of $\boldsymbol{c}$).

As shown in the green box in Fig. 1, the CTC output contains multiple approximately correct results. Unlike other single-step NAT models with length prediction where many redundant symbols are included in the CTC output, the compact CTC output $\boldsymbol{c}$ of our AL-NAT model does not include any blank symbols and repetition tokens, etc. We also don't need to insert the end-of-sentence '<eos>' token and adopt the causal masking mechanism.

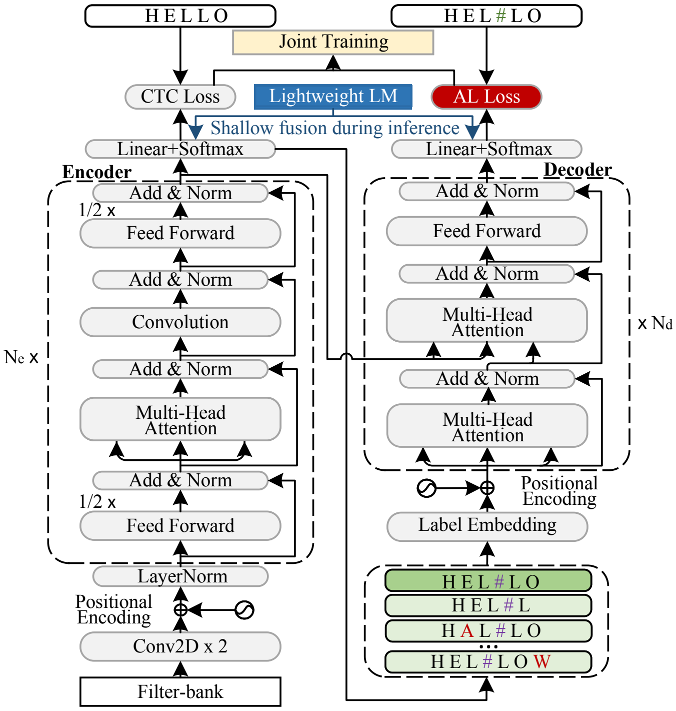

Fig. 1: Overview of the proposed alignment learning non-autoregressive transformer (AL-NAT) architecture.

The decoder takes the token sequence $\boldsymbol{c}$ and the encoder high-level representation $\mathcal{H}$ as input to generate the prediction sequences $\boldsymbol{y}$ in parallel. Our decoder consists of traditional transformer blocks [23]. The sine and cosine positional embedding proposed by [23] is applied to learn the positional information in the input sequence.

Traditional single-step NAT models employ CE loss [14, 16] in addition to the CTC loss [16] to force the CTC output $\boldsymbol{c}$ to be close to the target sequence y. Different from that, we define a novel alignment learning loss, denoted as AL Loss, in addition to the CTC loss [16], to joint optimize our AL-NAT. We will introduce the details of the AL Loss in Section 2.2.

2.2. Alignment learning loss

The proposed alignment learning loss seeks to find an optimal path to align the input $\boldsymbol{c}$ and target $\boldsymbol{y}$ that hold different lengths.

To this end, we first design a cost matrix $\Delta(\boldsymbol{c}, \boldsymbol{y})$ between input sequences $\boldsymbol{c}$ and target sequences $\boldsymbol{y}$.

$$\Delta(\boldsymbol{c}, \boldsymbol{y}) = [\delta(c_k, y_l)]_{kl} \in \mathbb{R}^{K \times L} \tag{2}$$

where $\delta(c_k, y_l)$ means a cost function to measure the alignment cost between the input token c_k and the target token y_l. Note that the value of $\delta(c_k, y_l)$ is a negative log-probability value obtained from the output of the decoder for each (k, l) pair.

In theory, there exists multiple eligible alignment warp paths from (c_1, y_1) to (c_K, y_L) in the cost matrix $\Delta(\boldsymbol{c}, \boldsymbol{y})$. We therefore defined A as a binary alignment matrix: $A_{kl} = 1$ if input c_k is labeled as y_l and $A_{kl} = 0$ otherwise. For each eligible alignment warp path, the inner product $\langle A, \Delta(\boldsymbol{c}, \boldsymbol{y})\rangle$ between the binary alignment matrix A and the cost matrix $\Delta(\boldsymbol{c}, \boldsymbol{y})$ represents the total cost of this path.

Our goal is to minimize the cost of these eligible alignment warp paths to find the best alignment path:

$$A^* = \arg\min_{A \in \mathcal{A}} \langle A, \Delta(\boldsymbol{c}, \boldsymbol{y}) \rangle \tag{3}$$

where $\mathcal{A}$ is a set of eligible binary alignment matrices.

Among many eligible alignments, the training goal of the alignment learning loss function is to optimize the model to minimize the cost of the optimal alignment:

$$\mathcal{L}_{\mathrm{AL}} = \min{}_{\gamma}\{\langle A, \Delta(\boldsymbol{c}, \boldsymbol{y}) \rangle, A \in \mathcal{A}\} \tag{4}$$

where the generalized $\min_\gamma\{\}$ operator is formulated as equation (5) with a smoothing parameter $0 < \gamma \leq 1$:

$$\min{}_{\gamma}\{a_1, ..., a_n\} = -\gamma \log \sum_{i=1}^{n} e^{-a_i/\gamma} \tag{5}$$

where a_i is the cost of the i *th* element in the alignment path. n is the number of alignment path elements.

Since the alignment process may yield monotonic one-to-one, one-to-many, and many-to-one results at the label level. We therefore insert an identifier "#" between the input sequence and output sequence of consecutively repeated labels to avoid error merging to a single label during inference. As shown in Fig.1, double "L" in "HELLO", "HELL", etc. are segmented with the identifier symbol "#" to avoid error merging to a single "L".

The final objective function of our AL-NAT overall architecture is defined as the weighted sum of the CTC loss on the encoder side and AL loss on the decoder side:

$$\mathcal{L}_{joint} = \lambda \mathcal{L}_{CTC} + (1 - \lambda)\mathcal{L}_{AL} \tag{6}$$

where λ is a scaling factor between the two loss terms.

2.3. Inference process with additional language model

In this section, we will introduce how the additional LM works in the inference stage.

As shown in the blue module of Fig. 1, we add a pretrained language model to both the encoder and decoder side, since the conditional independence assumption of the CTC cannot model the temporal context relationship information within the CTC output tokens and the NAT model cannot model the temporal dependencies between the output tokens.

Note that we adopt a lightweight 3-gram language model [24], rather than a large transformer language model [25], to avoid putting too much computational burden on the model and thus slowing down inference. During inference, the word-based beam search method applies beam pruning and token skipping operations to facilitate fast decoding [26]. Token skip means that tokens below the preset probability value will be skipped unless they are argmax value. We use $\mathrm{LM_{enc}}$ to indicate the language model attached to the encoder side, while use $\mathrm{LM_{dec}}$ to indicate that of the encoder side. $\mathrm{LM_{enc}}$ and $\mathrm{LM_{dec}}$ share the same parameter from the pretrained language model.

By fusing the $\mathrm{LM_{enc}}$, the CTC output $\boldsymbol{c}$ prediction process of the encoder is redefined as follows:

$$\boldsymbol{c} = \arg\max_{c}(\log P(c|\mathbf{x}; \theta_{enc}) + \alpha \log P_{LM_{enc}}(c|\mathbf{x})) \tag{7}$$

where $P_{LM_{enc}}(c|\mathbf{x})$ means the output probability of $\mathrm{LM_{enc}}$ and α means a weight factor.

Similar to Eq. (10), we can also reformulate the calculation of decoder armed with $\mathrm{LM_{dec}}$ as follows.

$$\boldsymbol{y} = \arg\max_{y}(\log P(y|\boldsymbol{c}; \theta_{dec}) + \alpha \log P_{LM_{dec}}(y|\boldsymbol{c})) \tag{8}$$

where θ_{dec} means the decoder model parameter.

3. EXPERIMENTS

3.1. Datasets

We report the performance on the English (LibriSpeech [20]) and Mandarin (Aishell1 [21]) respectively. The LibriSpeech dataset contains about 960 hours of read speech training data. The development and test data sets are come with its own and are split into "clean" and "other" subsets. The Aishell1 dataset includes about 150 hours of speech for training, about 18 hours of speech for development, and about 10 hours of speech for test.

3.2. Experimental setup

The input acoustic feature is 80-dimensional log-mel filter-bank (FBK) energies extracted by Kaldi toolkit [27]. The frame length is 25ms with a 10ms shift. Our AL-NAT contains 12 conformer block layers in the encoder and 6 transformer block layers in the decoder for the LibriSpeech experiments. We follow two settings, which are small and large, to build two AL-NAT models: AL-NAT (small) and AL-NAT (large). For AL-NAT (small), the dimension of the self-attention layer d_{model} was 256, and the inner-layer in fully connected feed-forward network d_{ff} has 1024 hidden units. We set the head number d_h of MHA to 4. For AL-NAT (large), parameter of the d_{model} = 512, d_{ff} = 2048, d_h = 8.

For LibriSpeech, the standard 3-gram language model (LM) based on the official text corpus [1] and trained using the KenLM toolkit [24] is employed for decoding. The parameters of beam search decoding are set as LM weight of 0.5, beam width of 28, pruning size of 10 and token skipping size of 5. We used a 10k subword vocabulary based on the Byte Pair Encoding (BPE) algorithm [28] as the modeling unit. For Aishell1, we use the AL-NAT (large) model configuration, but the number of conformer block layers in the encoder is reduced to 6. 4230 Chinese characters obtained from the training set are used as the modeling unit. We did not use any language models in the decoding phase.

In the training phase, layer normalization is applied before and dropout with a rate of 10% after each MHA and FNN layer. Perturbation [29] and SpecAugment [30] are applied to achieve data augmentation during all model training. Adam [31] optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$ was used for training with Noam learning rate schedule [23]. Warmup steps and learning rate constants were set to 25000 and 5.0, respectively. The batch contains up to 24,000 frames for all tasks. The hyper-parameter γ in Eq. (5) is set to 0.001 with empirical. We trained our models on NVIDIA Tesla P40 GPUs and the models were trained for 50 epochs for LibriSpeech tasks and 60 epochs for Aishell1 tasks. All models are implemented using the $neural\_sp$ [2] toolkit.

3.3. Experimental results

3.3.1. Comparison results of various scaling factor (λ)

Among the hyper-parameters that affect the AL-NAT system performance, we would like to discuss the scaling factor λ in Eq. (6), which balances the contributions between CTC loss and our AL

[1] http://www.openslr.org/resources/11/librispeech-lm-norm.txt.gz

[2] https://github.com/hirofumi0810/neural_sp

Table 1: Comparison results of various scaling factors λ of the proposed AL-NAT model in terms of character error rate (CER).

λ	CER (%)			
	Encoder CTC search		Decoder AL search	
	dev	test	dev	test
0.1	9.7	10.8	5.4	6.0
0.3	8.6	9.4	5.4	5.8
0.5	7.0	7.4	5.3	5.6
0.8	**5.7**	**6.3**	**4.9**	**5.3**
0.9	5.9	7.0	5.4	5.5

loss. We report the performance on Aishell1 dataset for 5 systems with $\lambda \in \{0.1, 0.3, 0.5, 0.8, 0.9\}$.

Table 1 presents the results in terms of character error rate (CER) for encoder and decoder output respectively. We find that our AL-NAT model achieves the best performance when $\lambda = 0.8$ for both Encoder CTC search and Decoder AL search. A higher or lower value of λ does not lead to better performance. A similar trend also appeared in the fourth and fifth columns of Table 1. In the following experiments, we set $\lambda = 0.8$ to build our AL-NAT model.

3.3.2. Comparison results on Aishell1 and Librispeech

We conduct experiments on Aishell1 and Librispeech datasets to validate our AL-NAT model in terms of recognition error rate and inference speed.

Following the [17, 18], we first report the CER and RTF results of Aishell1 for different models without LM in Table 2. As shown in the third and fourth columns of Table 2, we observe that our proposed AL-NAT model outperforms all baselines and achieves the best performance on both development and test set. As shown in the fifth column of Table 2, we find that our AL-NAT model achieves a remarkable RTF value of 0.005 and outperforms all baseline systems, except than BERT-LASO and LASO. We find that the RTF values of *LASO* and *BERT-LASO* models are obtained by testing under NVIDIA RTX 2080Ti GPU. Their machine has a faster inference speed [3] compared to NVIDIA Tesla P40 GPU that used in our work, but our RTF value only differs from their results by 0.001, which is encouraging.

We further report the WER and RTF results of Librispeech for different models in Table 3. Note that the results of Librispeech can be categorized into: 1) without LM, and 2) with LM. The third-eighth rows of Table 3 report the results of different models without LM. We can see that the WER value of our AL-NAT (large) w/o LM model is comparable to the state-of-the-art baseline Improved CASS-NAT, while the WER value of AL-NAT (small) w/o LM model is slightly worse than the baseline due to the small amount of parameters. Note that the RTF values of our models are 0.006 and 0.007, which perform about 2x faster than *Improved CASS-NAT* of 0.014. Even though their machine NVIDIA Tesla V100 GPU outperforms our NVIDIA Tesla P40 GPU.

Table 2: Comparison of the CER and RTF on Aishell1. RTFs of previous work are obtained directly from the authors report.

System	#Param	CER (%)		RTF
		dev	test	
ST-NAT [14]	-	6.9	7.7	0.006
ST-NAT+LM [14]	-	6.4	7.0	0.029
LASO [18]	80.0M	5.8	6.4	0.004
CASS-NAT [16]	-	5.3	5.8	0.011
BERT-LASO [19]	80.0M	5.2	5.8	0.004
Improved CASS-NAT [17]	-	4.9	5.4	0.023
AL-NAT	**71.3M**	**4.9**	**5.3**	**0.005**

Table 3: Comparison of WER and RTF on LibriSpeech. RTFs of previous work are obtained directly from the authors report.

System	WER (%)				RTF
	Dev		Test		
	clean	other	clean	other	test-clean
Without LM					
Imputer [8]	-	-	4.0	11.1	-
Align-refine [13]	-	-	3.6	9.0	-
CASS-NAT-ESA [16]	3.7	9.2	3.8	9.1	0.010
Improved CASS-NAT [17]	2.8	7.3	3.1	7.2	0.014
AL-NAT (small) w/o LM	3.4	8.7	3.6	8.7	0.006
AL-NAT (large) w/o LM	2.9	7.4	3.2	7.4	0.007
With LM					
CASS-NAT-ESA [16]	3.3	8.0	3.3	8.1	-
Improved CASS-NAT [17]	-	-	2.8	6.5	0.188
AL-NAT (small) w/o $\mathrm{LM}_{\mathrm{dec}}$	3.2	8.2	3.5	8.4	0.008
AL-NAT (small) w/o $\mathrm{LM}_{\mathrm{enc}}$	2.9	7.5	3.3	7.7	0.007
AL-NAT (small)	**2.8**	**7.2**	**3.1**	**7.4**	**0.010**
AL-NAT (large) w/o $\mathrm{LM}_{\mathrm{dec}}$	2.7	7.1	3.0	7.2	0.009
AL-NAT (large) w/o $\mathrm{LM}_{\mathrm{enc}}$	2.5	6.4	2.9	6.5	0.008
AL-NAT (large)	**2.4**	**6.2**	**2.8**	**6.3**	**0.011**

To validate the performance of the additional language models plugged on encoder and decoder side. The tenth-seventeenth rows of Table 3 report the results of different models with LM. $\mathrm{w/oLM}_{\mathrm{enc}}$ and $w/o\mathrm{LM}_{\mathrm{dec}}$ mean that we remove the language models on encoder and decoder sides respectively. Compared our AL-NAT models with the two baselines, we observe that the *AL-NAT (small)* and *AL-NAT (large)* models perform lower WER with smaller RTF. Specifically, compared to the *Improved CASS-NAT* system using transformer-based LM, the WER of the AL-NAT (large) model on test-other perform 0.2 lower than that of *Improved CASS-NAT*, with about 17x faster in terms of RTF. We believe that the lightweight language models contribute to learning the temporal context knowledge for decoding to further improve performance.

Furthermore, by comparing AL-NAT (small) w/o $\mathrm{LM}_{\mathrm{enc}}$, AL-NAT (small) w/o $\mathrm{LM}_{\mathrm{dec}}$ and AL-NAT (small) three groups of experiments. We find that removing the language model on encoder or decoder side respectively resulted in performance degradation of WER, although they perform fast decoding speed. The same experimental results have also been verified in L-NAT (large) w/o $\mathrm{LM}_{\mathrm{enc}}$, AL-NAT (large) w/o $\mathrm{LM}_{\mathrm{dec}}$ and AL-NAT (large) three groups of experiments. This further proves the validity of the language model.

4. CONCLUSION

This paper proposed a novel single-step AL-NAT training architecture for end-to-end speech recognition, termed as AL-NAT. We defined an alignment learning loss to learn the accurate alignment for the source-target sequence pairs. During inference, we added a lightweight language model to both the encoder and decoder side to improve the overall performance of the NAT model. The results show that our model outperforms all baseline models and achieves accurate and fast recognition performance.

[3] https://technical.city/zh/video/Tesla-P40-vs-GeForce-RTX-2080-Ti

5. REFERENCES

[1] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *ICASSP*, 2016, pp. 4960–4964.

[2] Suyoun Kim, Takaaki Hori, and Shinji Watanabe, “Joint ctc-attention based end-to-end speech recognition using multi-task learning,” in *ICASSP*, 2017, pp. 4835–4839.

[3] Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani, “State-of-the-art speech recognition with sequence-to-sequence models,” in *ICASSP*, 2018, pp. 4774–4778.

[4] Jinyu Li, Yu Wu, Yashesh Gaur, Chengyi Wang, Rui Zhao, and Shujie Liu, “On the comparison of popular end-to-end models for large scale speech recognition,” in *Interspeech*, 2020, pp. 1–5.

[5] Shiyu Zhou, Linhao Dong, Shuang Xu, and Bo Xu, “Syllable-based sequence-to-sequence speech recognition with the transformer in mandarin chinese,” in *Interspeech*, 2018, pp. 791–795.

[6] Linhao Dong, Shuang Xu, and Bo Xu, “Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition,” in *ICASSP*, 2018, pp. 5884–5888.

[7] Yongqiang Wang, Abdelrahman Mohamed, Due Le, Chunxi Liu, Alex Xiao, Jay Mahadeokar, Hongzhao Huang, Andros Tjandra, Xiaohui Zhang, Frank Zhang, Christian Fuegen, Geoffrey Zweig, and Michael L. Seltzer, “Transformer-based acoustic modeling for hybrid speech recognition,” in *ICASSP*, 2020, pp. 6874–6878.

[8] William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly, “Imputer: Sequence modelling via imputation and dynamic programming,” in *ICML*, 2020, pp. 1403–1413.

[9] Yosuke Higuchi, Shinji Watanabe, Nanxin Chen, Tetsuji Ogawa, and Tetsunori Kobayashi, “Mask ctc: Non-autoregressive end-to-end asr with ctc and mask predict,” in *Interspeech*, 2020, pp. 3655–3659.

[10] Yosuke Higuchi, Hirofumi Inaguma, Shinji Watanabe, Tetsuji Ogawa, and Tetsunori Kobayashi, “Improved mask-ctc for non-autoregressive end-to-end asr,” in *ICASSP*, 2021, pp. 8363–8367.

[11] Nanxin Chen, Shinji Watanabe, Jesús Villalba, and Najim Dehak, “Non-autoregressive transformer automatic speech recognition,” *Signal Processing Letters*, 2020.

[12] Yuya Fujita, Shinji Watanabe, Motoi Omachi, and Xuankai Chang, “Insertion-based modeling for end-to-end automatic speech recognition,” in *Interspeech*, 2020, pp. 3660–3664.

[13] Ethan A. Chi, Julian Salazar, and Katrin Kirchhoff, “Align-refine: Non-autoregressive speech recognition via iterative realignment,” in *NAACI*, 2021, pp. 1920–1927.

[14] Zhengkun Tian, Jiangyan Yi, Jianhua Tao, Ye Bai, Shuai Zhang, and Zhengqi Wen, “Spike-triggered non-autoregressive transformer for end-to-end speech recognition,” in *Interspeech*, 2020, pp. 5026–5030.

[15] Nanxin Chen, Piotr Żelasko, Laureano Moro-Velázquez, Jesús Villalba, and Najim Dehak, “Align-denoise: Single-pass non-autoregressive speech recognition,” in *Interspeech*, 2021, pp. 3770–3774.

[16] Ruchao Fan, Wei Chu, Peng Chang, and Jing Xiao, “Cass-nat: Ctc alignment-based single step non-autoregressive transformer for speech recognition,” in *ICASSP*, 2021, pp. 5889–5893.

[17] Ruchao Fan, Wei Chu, Peng Chang, Jing Xiao, and Abeer Alwan, “An improved single step non-autoregressive transformer for automatic speech recognition,” in *Interspeech*, 2021, pp. 3715–3719.

[18] Ye Bai, Jiangyan Yi, Jianhua Tao, Zhengkun Tian, Zhengqi Wen, and Shuai Zhang, “Listen attentively, and spell once: Whole sentence generation via a non-autoregressive architecture for low-latency speech recognition,” in *Interspeech*, 2020, pp. 3381–3385.

[19] Ye Bai, Jiangyan Yi, Jianhua Tao, Zhengkun Tian, Zhengqi Wen, and Shuai 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.

[20] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, “Librispeech: an asr corpus based on public domain audio books,” in *ICASSP*, 2015, pp. 5206–5210.

[21] Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng, “Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline,” in *O-COCOSDA*, 2017, pp. 1–5.

[22] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al., “Conformer: Convolution-augmented transformer for speech recognition,” in *ICASSP*, 2020, pp. 5036–5040.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, “Attention is all you need,” in *ICONIP*, 2017, pp. 5998–6008.

[24] Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn, “Scalable modified kneser-ney language model estimation,” in *ACL*, 2013, pp. 690–696.

[25] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov, “Transformer-xl: Attentive language models beyond a fixed-length context,” in *ACL*, 2020, pp. 2978–2988.

[26] Harald Scheidl, Stefan Fiel, and Robert Sablatnig, “Word beam search: A connectionist temporal classification decoding algorithm,” in *ICFHR*, 2018, pp. 253–258.

[27] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukáš Burget, Ondřej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlíček, Yanmin Qian, and Petr Schwarz, “The kaldi speech recognition toolkit,” in *ASRU*. IEEE, 2011.

[28] Rico Sennrich, Barry Haddow, and Alexandra Birch, “Neural machine translation of rare words with subword units,” in *ACL*, 2016, pp. 7–12.

[29] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, “Audio augmentation for speech recognition,” in *Interspeech*, 2015, pp. 3586–3589.

[30] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin Dogus Cubuk, and Quoc V. Le, “Specaugment: A simple data augmentation method for automatic speech recognition,” in *Interspeech*, 2019, pp. 2613–2617.

[31] Diederik P. Kingma and Jimmy Lei Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2015.