IMPROVING NON-AUTOREGRESSIVE SPEECH RECOGNITION WITH AUTOREGRESSIVE PRETRAINING

Yanjia Li, Lahiru Samarakoon, Ivan Fung

Fano Labs, Hong Kong

ABSTRACT

Autoregressive (AR) automatic speech recognition (ASR) models predict each output token conditioning on the previous ones, which slows down their inference speed. On the other hand, non-autoregressive (NAR) models predict tokens independently and simultaneously within a constant number of decoding iterations, which brings high inference speed. However, NAR models generally have lower accuracy than AR models. In this work, we propose AR pretraining to the NAR encoder to reduce the accuracy gap between AR and NAR models. The experiment results show that our AR-pretrained MaskCTC reaches the same accuracy as AR Conformer on Aishell-1 (both 4.9% CER) and reduce the performance gap with AR Conformer on LibriSpeech by relatively 50%. Moreover, our AR-pretrained MaskCTC only needs single decoding iteration, which reduces inference time by 50%. We also investigate multiple masking strategies in training the masked language model of MaskCTC.

***Index Terms*—** Non-autoregressive ASR, autoregressive ASR, MaskCTC, end-to-end speech recognition

1. INTRODUCTION

The performance of automatic speech recognition (ASR) has greatly benefits from the rapid development of sequence-to-sequence modeling. Many autoregressive (AR) ASR models [1, 2, 3, 4] from recent studies have achieved superior performance in ASR tasks. However, AR models predict each token one by one depending on the previously predicted ones. This conditional dependency makes inference time proportional to output sequence length, which leads to slow inference speed.

In contrast to AR ASR models, non-autoregressive (NAR) ASR models are able to generate output tokens in parallel within a constant number of decoding iterations. CTC generates frame-wise input-output alignments based on the conditional independence assumption between output tokens [5]. However, this assumption degrades the performance of CTC. Some recent models [6, 7, 8, 9, 10, 11] relax this assumption and make token predictions based on intermediate decoding results. MaskCTC [9] masks low-confidence tokens in a CTC output sequence and re-predicts the masked tokens based on the observed high-confidence tokens. Align-Refine [6] refines the CTC alignments iteratively using the non-causal Transformer as the decoder.

In general, NAR models are outperformed by AR models in terms of accuracy. To reduce this performance gap, some methods are proposed to better handle the dependencies among tokens and give more accurate length prediction. [12] has utilized pre-trained wav2vec2.0 [13] and BERT [14] model to improve the recognition performance of MaskCTC. [15] has designed a dynamic length prediction algorithm to enable flexible length prediction of output sequence for MaskCTC. However, those methods either require external acoustic and language models or have minor improvement on the model performance. On the other hand, [16] has proposed knowledge transfer and distillation from AR models to improve MaskCTC. But it makes model training complicated and increases the computational complexity.

In this work, we propose AR pretraining to NAR encoder, aiming to reduce the performance gap between AR models and NAR models. To demonstrate the effectiveness of our methods, we improve MaskCTC by using pretrained Conformer encoder from AR models to initialize MaskCTC during training. To alleviate the train-test condition mismatch, we also propose to use CTC outputs as the decoder input directly during MaskCTC training with different masking strategies. The proposed AR-pretrained MaskCTC reaches the same performance as AR models on Aishell-1 benchmark and greatly reduces its gap with AR models on LibriSpeech.

2. RELATED WORK

In this section, we first review the autoregressive (AR) ASR and the non-autoregressive (NAR) ASR. We briefly introduce CTC and MaskCTC as examples of NAR models.

We define end-to-end (E2E) ASR as a sequence mapping from an input sequence $X = (x_1, x_2, \ldots, x_T)$ of length T, to a output sequence $Y = (y_1, y_2, \ldots, y_L)$ of length L, where x_t is the acoustic feature at time t and y_l is the output token at position l.

2.1. Autoregressive (AR) ASR

AR ASR predicts each output token y_l one by one in the left-to-right order by conditioning on previously generated tokens $y_{<l}$ as shown below:

$$P(Y|X) = \prod_{l=1}^{L} P(y_l|y_{<l}, X) \tag{1}$$

2.2. Non-autoregressive (NAR) ASR

NAR ASR predicts output tokens in parallel by conditioning on intermediate decoding results and input X.

2.2.1. Connectionist Temporal Classification (CTC)

CTC generates frame-level alignments between input X and output alignment $A = \{a_t \in \mathcal{V} \cup \epsilon | t = 1, 2, \ldots, T\}$, where $\mathcal{V}$ is the vocabulary set and ϵ is a blank symbol. It bases on the conditional independence assumption among all output tokens and predicts the probability $P(Y|X)$ by combining all possible paths as:

$$P_{CTC}(Y|X) = \sum_{A \in \beta^{-1}(Y)} P(A|X) = \sum_{A \in \beta^{-1}(Y)} \prod_{t=1}^{T} P_{CTC}(a_t|X) \quad (2)$$

where $\beta^{-1}(Y)$ represents all possible alignments compatible with Y.

To reach fast convergence and robust alignment, we use the hybrid CTC/attention architecture to train AR models [17]. The AR objective is also known as the attention objective under this hybrid architecture. The joint training objective is defined as follows with a combination of Eq.(1) and Eq.(2):

$$L_{AR} = \lambda \log P_{CTC}(Y|X) + (1-\lambda) \log P_{att}(Y|X) \quad (3)$$

where λ satisfies $0 \leq \lambda \leq 1$.

2.2.2. MaskCTC

MaskCTC adopts an encoder-decoder architecture based on Transformer blocks [9, 1]. It applies CTC objective to encoder outputs and uses the conditional masked language model (CMLM) as the decoder [18]. During training, a subset of ground truth tokens Y_{mask} is randomly masked by a special token $\langle$MASK$\rangle$. The CMLM decoder is trained to predict Y_{mask} based on the rest unmasked tokens Y_{obs} as follows:

$$P_{cmlm}(Y_{mask}|Y_{obs}, X) = \prod_{y \in Y_{mask}} P(y|Y_{obs}, X) \quad (4)$$

MaskCTC is trained with a joint CTC and mask-predict objective as formulated below:

$$\mathcal{L}_{NAR} = \lambda \log P_{CTC}(Y|X) + (1-\lambda) \log P_{cmlm}(Y_{mask}|Y_{obs}, X) \quad (5)$$

During inference, we first obtain a CTC output through greedy decoding. After collapsing the repetitive tokens and removing the blank symbol, we mask tokens with posterior CTC probabilities lower than the probability threshold P_{thres}. The remaining unmasked tokens are passed to the CMLM decoder to generate the masked tokens. The CMLM decoder gradually refills the CTC output sequence within K decoding iterations, given all the high-confidence tokens as bi-directional context.

Table 1. *The character error rates (CER) (%) of CTC greedy search and decoder output of different AED-based AR and NAR models on Aishell-1 test set*

Model	Greedy CTC(%)	Decoder(%)
–Autoregressive		
AR Conformer	**5.3**	**4.9**
–Non-autoregressive		
Conformer CTC (16-layers)	6.6	-
Conformer MaskCTC	5.4	5.2

3. AUTOREGRESSIVE PRETRAINING FOR NON-AUTOREGRESSIVE ASR

In general, NAR ASR models are outperformed by AR models in terms of accuracy. Multiple factors may contribute to this performance difference. First, CTC makes the conditional independence assumption among output tokens to predict all tokens in parallel. To reduce the performance gap with AR models, MaskCTC relaxes the conditional independence assumption and makes token predictions based on a set of observed tokens from the intermediate output sequence. Second, NAR models cannot adjust output sequence length after the initial prediction. For example, MaskCTC fixes the length of an output sequence with that of the CTC output, making it hard for the CMLM decoder to correct deletion and insertion errors in the CTC output.

Since the encoder is common in both AR and NAR models, it is worth investigating the effect of the encoder on the performance gap between NAR and AR models. Given that we use the hybrid CTC/attention architecture for all AR and NAR models, we can use CTC outputs with greedy search to evaluate their encoders. As shown in Table 1, we find that the CTC of AR Conformer performs better than that of Conformer MaskCTC and pure CTC model. The CER of greedy CTC outputs of AR Conformer is 5.3%, which is 0.1% better than that of MaskCTC and 1.3% better than that of the pure CTC model. It implies that the encoder is trained more efficiently under the AR objective.

Based on this observation, we propose AR pretraining for NAR speech recognition in order to improve the performance of NAR models. With better initialization using pretrained encoder from the AR counterpart, we can train the encoder of MaskCTC to be more powerful to capture context information from sequential data. It can help improve the model performance by compensating the relaxed conditional independence assumption made by MaskCTC. Furthermore, since encoder is a common component, other NAR models, such as CTC[5] and Align-Refine [6], can also benefit from our AR pretraining method.

3.1. Training CMLM decoder with CTC outputs

MaskCTC uses ground truth as decoder input during training while using CTC greedy output sequence during inference. This is because, at the early stage of training, CTC contains so many errors that makes it difficult to train the CMLM decoder. However, this mismatch between training and testing conditions can degrade the model performance. According to Table 1, the CTC of AR Conformer has a CER of 5.3% using greedy search. It suggests that the CTC greedy output from a pretrained AR model is accurate and reliable to a considerable extent. Therefore, we propose to use the CTC greedy output as the decoder input directly to alleviate the train-test condition mismatch. During training, if the length of a CTC output sequence equals the length of the corresponding ground truth, we will use the CTC output as the CMLM decoder input. Otherwise we keep using the ground truth as the CMLM decoder input.

3.2. Masking strategies for CTC outputs

We propose two masking strategies to process CTC outputs as decoder inputs during training:

- **Confidence Masking:** Same as MaskCTC inference, we first obtain a CTC output sequence by greedy search and then mask a subset of CTC tokens that have confidence scores lower than P_{thres} with the special symbol $\langle$MASK$\rangle$. We set

Table 2. *Model comparison on CERs on Aishell-1*

Model	# of Params(M)	Greedy CTC(%)	Decoder(%)
–Autoregressive			
AR Conformer	46.25	5.3	**4.9**
–Non-autoregressive			
Conformer CTC (16 layers)	45.09	6.6	-
A-FMLM [11]	-	-	6.7
LASO-big [19]	-	-	6.4
CASS-NAT [7]	29.7	-	5.8
NAT-UBD [20]	29.7	-	5.5
CTC-enhanced [10]	29.7	-	5.9
AL-NAT [8]	71.3	-	5.3
Transformer MaskCTC (K=5)	30.4	5.9	5.4
Conformer MaskCTC (K=5)	46.25	5.4	5.2
+ AR Pretraining (K=5)	46.25	5.0	**4.9**
+ AR Pretraining (K=1)	46.25	5.0	**4.9**
+ CTC & Confidence Masking	46.25	**4.9**	**4.9**
+ CTC & Random Masking	46.25	**4.9**	**4.9**

a strict threshold P_{thres} to make more tokens masked during training, which helps to get fast convergence.

- **Random Masking:** For each CTC output sequence of length L, we decide the number of masked tokens N using a uniform distribution $U(0, L)$ and then randomly replace N CTC tokens with symbol $\langle \text{MASK} \rangle$ from the output sequence.

4. EXPERIMENTS

4.1. Datasets

We evaluate our proposed method on Aishell-1 Mandarin corpus (178 hours) [21] and LibriSpeech English corpus (960 hours) [22]. For Aishell-1, a vocabulary size of 4,233 Chinese characters is used. For LibriSpeech, a vocabulary size of 5,000 sub-word units is used.

4.2. Experimental setup

For ASR models in all tasks, we train Conformer encoders of 12 blocks and Transformer decoders of 6 blocks. In Aishell-1 experiments, each attention layer in the encoder-decoder architecture consists of 256 hidden units and 4 attention heads, and each feed-forward layer has 2048 hidden units. The depth-wise convolutional layers of Conformer have a kernel size of 15. In LibriSpeech experiments, each self-attention layer has 512 hidden units and 8 attention heads. The depth-wise convolutional layers of Conformer have a kernel size of 31.

We use 80 dimensional mel-scale filterbank features for all the models. SpecAugment [23] and speed perturbation [24] is applied to the input data. Network parameters are trained with 50 epochs for AR models and 100 epochs for NAR models using Adam optimizer [25]. For AR models, we set warm-up steps to be 30,000 for Aishell-1 and 40,000 for LibriSpeech. For NAR models, we set warm-up steps to be 15,000 for both datasets. The hybrid CTC/attention architecture [17] with a CTC weight of 0.3 are used for all models during training.

The inference models are all generated by averaging the model parameters of 10 epochs with the best validation accuracy. Some detailed configurations for specific evaluated models are provided in Section 4.3.

All the models are trained using ESPnet recipes [26]. We evaluate the model performance without external language models.

Table 3. *CER and RTF comparison between AR and NAR models on Aishell-1*

Model	CER(%)	RTF
–Autoregressive		
AR Conformer (beam size=20)	4.9	0.779
–Non-autoregressive		
MaskCTC (K=5)	5.4	0.012
Conformer CTC (Greedy Search)	6.6	0.003
AR-pretrained Conformer MaskCTC (Greedy CTC)	5.0	0.003
AR-pretrained Conformer MaskCTC (K=1)	4.9	0.006

4.3. Evaluated models

In our experiments, we compare the performances of three baseline models: AR Conformer, Conformer CTC and MaskCTC. Detailed model configurations for each baseline are given below.

- **AR Conformer:** An AR model trained with the joint CTC-attention objective in Eq.(1). During inference, we use a default beam size of 20 for Aishell-1 and beam size of 60 for LibriSpeech as suggested in [26].
- **Conformer CTC:** An NAR model simply trained with the CTC objective. We use a 16-layer Conformer encoder to match the model size with others. We apply CTC greedy search for inference.
- **Conformer MaskCTC:** An NAR model trained with Eq.(5). As defined in Section 2.2.2, we set the number of decoding iteration K to 5 and threshold P_{thres} to 0.9 during inference. To obtain CTC greedy output, we set P_{thres} to 0.0. With AR pretraining, we use the encoder and CTC from the above Conformer model for initialization. When using confidence masking, we set P_{thres} to 0.99 for training.

5. RESULTS

In Table 2, we summarize the published results reported on Aishell-1. It can be observed that a clear CER performance gap is present

Table 4. *Model comparison on word error rates (WERs) for LibriSpeech.*

Model	# of Params(M)	Enc. Layers	Dec. Layers	Greedy CTC(%)		Decoder(%)	
				test-clean	test-other	test-clean	test-other
–Autoregressive							
AR Conformer	116.15	12	6	2.8	6.5	**2.3**	**5.4**
–Non-autoregressive							
Transformer CTC [27]	-	16	-	4.6	13.0	-	-
Conformer CTC [28]	115.7	17	-	2.7	5.9	-	-
Imputer [27]	-	16	-	-	-	4.0	11.1
Align-Refine [6]	-	12	6	4.6	11.5	3.6	9.0
CASS-NAT [7]	-	12	-	-	-	3.8	9.1
AL-NAT [8]	-	12	6	-	-	3.2	7.4
Conformer MaskCTC [16]	115.0	12	6	-	-	4.1	10.2
+ Knowledge Transfer [16]	115.0	12	6	-	-	3.3	7.8
Conformer MaskCTC (K=5)	116.15	12	6	2.9	6.5	2.8	6.2
+ AR Pretraining (K=5)	116.15	12	6	**2.6**	**5.8**	**2.5**	**5.7**
+ AR Pretraining (K=1)	116.15	12	6	**2.6**	**5.8**	**2.5**	**5.7**
+ CTC & Confidence Masking	116.15	12	6	**2.6**	5.9	**2.5**	5.8
+ CTC & Random Masking	116.15	12	6	**2.6**	5.9	**2.5**	**5.7**

Table 5. CER and RTF comparison between AR and NAR models on LibriSpeech test-other

Model	CER(%)	RTF
–Autoregressive		
AR Conformer (beam size=60)	5.4	6.848
–Non-autoregressive		
Conformer CTC (Greedy Search)	5.9	0.004
AR-pretrained Conformer MaskCTC (Greedy CTC)	5.8	0.004
AR-pretrained Conformer MaskCTC (K=1)	5.7	0.009

between AR and NAR baseline models. The AR model has the best CER performance of 5.3% and 4.9% with CTC and decoder output respectively. Compared with other reported NAR models, our Conformer MaskCTC is a strong baseline, which gives a CER of 5.4% and 5.2% using CTC and decoder output respectively. With our proposed AR pre-training, it outperforms the AR model by 0.3% with CTC output, and reaches the same CER performance with decoder output when $K = 1$ is used. We do not observe any additional gain when $K > 1$ is used with our proposed method, possibly due to the more accurate CTC outputs with AR pretraining. This demonstrates that our proposed method is effective in minimizing the CER performance gap between AR and NAR models.

The CER and RTF performance are summarized in Table 3. The RTF for the AR model is 0.779, which is the worst because each prediction is conditional on the previous output tokens. Due to the parallel token prediction nature of the NAR model, Conformer MaskCTC with $K = 5$ gives a better RTF of 0.012. However, it has a worse CER of 5.4% in comparison with 4.9% of the AR model. With our proposed AR pretraining, the Conformer MaskCTC achieves the same CER performance to the AR model even when $K = 1$ is used, while the RTF is $\sim$130$\times$ better. In addition, we achieve another 2$\times$ better RTF when CTC greedy decoding is used, while maintaining similar CER performance.

Next, we evaluate our proposed idea on LibriSpeech. In Table 4, the decoder WER of test-clean and test-other decrease from 2.8% to 2.5% and 6.2% to 5.7% respectively when AR pretraining is used, in comparison to our NAR baseline. Similar improvement can also be observed when greedy CTC decoding is used. This demonstrates that our proposed idea outperforms other NAR models, including a recent parallel work using knowledge transfer and distillation from AR model [16]. We summarize the CER and RTF performance among AR and NAR models for LibriSpeech in Table 5.

When training the CMLM decoder with CTC outputs, it only brings a little improvement to CER on Aishell-1, yet a slight degradation to WER on LibriSpeech. We hypothesise that the CTC outputs are close to the ground truth when AR pretraining is used, therefore it does not make a significant difference.

6. CONCLUSION

In this paper, we proposed AR pretraining to NAR speech recognition in order to reduce the performance gap between AR and NAR ASR models. We demonstrated the effectiveness of the proposed method by training MaskCTC with pretrained Conformer encoder from AR models. The experiment results on Aishell-1 and LibriSpeech showed that the AR-pretrained MaskCTC outperformed other NAR baseline models and reached competitive performance to AR Conformer models. It suggested that our AR pretraining method improved the encoder and compensated the relaxed conditional independence assumption used by MaskCTC. We also investigated the effect of using CTC outputs as decoder inputs for MaskCTC with two masking strategies. Given no significant difference from this method, we think it is sufficient to simply use ground truth as decoder inputs. Moreover, the AR-pretrained MaskCTC required only one decoding iteration to reach the best accuracy, which improved the inference latency considerably. Since the encoder is common to many E2E NAR ASR models, proposed AR pretraining can be applied to NAR models other than MaskCTC.

7. REFERENCES

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," 2017.

[2] Linhao Dong, Shuang Xu, and Bo Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5884–5888.

[3] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang, "Conformer:

Convolution-augmented transformer for speech recognition," 2020.

[4] Alex Graves, "Sequence transduction with recurrent neural networks," 2012.

[5] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA, 2006, ICML '06, p. 369–376, Association for Computing Machinery.

[6] Ethan A. Chi, Julian Salazar, and Katrin Kirchhoff, "Align-refine: Non-autoregressive speech recognition via iterative realignment," 2020.

[7] Ruchao Fan, Wei Chu, Peng Chang, and Jing Xiao, "Cassnat: 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)*, 2021, pp. 5889–5893.

[8] Yonghe Wang, Rui Liu, Feilong Bao, Hui Zhang, and Guanglai 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)*, 2022, pp. 8292–8296.

[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," 2020.

[10] Xingchen Song, Zhiyong Wu, Yiheng Huang, Chao Weng, Dan Su, and Helen Meng, "Non-autoregressive transformer asr with ctc-enhanced decoder input," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5894–5898.

[11] Nanxin Chen, Shinji Watanabe, Jesus Villalba, Piotr Zelasko, and Najim Dehak, "Non-autoregressive transformer for speech recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 121–125, 2021.

[12] Keqi Deng, Songjun Cao, Yike Zhang, and Long Ma, "Improving hybrid ctc/attention end-to-end speech recognition with pretrained acoustic and language model," 2021.

[13] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2020, NIPS'20, Curran Associates Inc.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio, Eds. 2019, pp. 4171–4186, Association for Computational Linguistics.

[15] Yosuke Higuchi, Hirofumi Inaguma, Shinji Watanabe, Tetsuji Ogawa, and Tetsunori Kobayashi, "Improved mask-ctc for non-autoregressive end-to-end asr," 2020.

[16] Xun Gong, Zhikai Zhou, and Yanmin Qian, "Knowledge transfer and distillation from autoregressive to non-autoregressive speech recognition," 2022.

[17] Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R. Hershey, and Tomoki 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.

[18] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer, "Constant-time machine translation with conditional masked language models," *CoRR*, vol. abs/1904.09324, 2019.

[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 Trans. Audio, Speech and Lang. Proc.*, vol. 29, pp. 1897–1911, jan 2021.

[20] Chuan-Fei Zhang, Yan Liu, Tian-Hao Zhang, Song-Lu Chen, Feng Chen, and Xu-Cheng Yin, "Non-autoregressive transformer with unified bidirectional decoder for automatic speech recognition," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 6527–6531.

[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," *CoRR*, vol. abs/1709.05522, 2017.

[22] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[23] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Interspeech 2019*. sep 2019, ISCA.

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

[25] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," 2014.

[26] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai, "Espnet: End-to-end speech processing toolkit," *CoRR*, vol. abs/1804.00015, 2018.

[27] William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, and Navdeep Jaitly, "Imputer: Sequence modelling via imputation and dynamic programming," 2020.

[28] Edwin G. Ng, Chung-Cheng Chiu, Yu Zhang, and William Chan, "Pushing the limits of non-autoregressive speech recognition," in *Interspeech 2021*. aug 2021, ISCA.