INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Listen Attentively, and Spell Once: Whole Sentence Generation via a Non-Autoregressive Architecture for Low-Latency Speech Recognition

Ye Bai[1,2]*, Jiangyan Yi*[1]*, Jianhua Tao*[1,2,3]*, Zhengkun Tian*[1,2]*, Zhengqi Wen*[1]*, Shuai Zhang*[1,2]

[1]NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing
[3]CAS Center for Excellence in Brain Science and Intelligence Technology
{ye.bai, jiangyan.yi, jhtao, zhengkun.tian, zqwen, shuai.zhang}@nlpr.ia.ac.cn

Abstract

Although attention based end-to-end models have achieved promising performance in speech recognition, the multi-pass forward computation in beam-search increases inference time cost, which limits their practical applications. To address this issue, we propose a non-autoregressive end-to-end speech recognition system called LASO (listen attentively, and spell once). Because of the non-autoregressive property, LASO predicts a textual token in the sequence without the dependence on other tokens. Without beam-search, the one-pass propagation much reduces inference time cost of LASO. And because the model is based on the attention based feedforward structure, the computation can be implemented in parallel efficiently. We conduct experiments on publicly available Chinese dataset AISHELL-1. LASO achieves a character error rate of 6.4%, which outperforms the state-of-the-art autoregressive transformer model (6.7%). The average inference latency is 21 ms, which is $1/50$ of the autoregressive transformer model.

Index Terms: speech recognition, sequence-to-sequence, non-autoregressive, transformer

1. Introduction

Attention based sequence-to-sequence (Seq2Seq) speech recognition systems have achieved promising performance these years [1, 2, 3]. In these models, an encoder encodes acoustic features into high-level representations. And a decoder is a conditional language model, which predicts the next token in terms of the previous tokens and the acoustic context. At inference stage, the decoder finds the most likely token sequence approximately with beam-search algorithm. This paradigm shows powerful ability for sequence generation. However, even with non-recurrent structures (transformers) for parallelization [4, 5], the autoregressive manner still affects inference speed.

Non-autoregressive Seq2Seq models were proposed for speeding up the inference of machine translation systems [6, 7, 8, 9]. These models also use an encoder-decoder architecture with attention mechanism. But they can predict all tokens in parallel rather than in step-by-step manner. It avoids multi-pass forward propagation of the decoder in beam-search, so the inference time cost is much reduced. However, the performances of these models fall behind the state-of-the-art autoregressive models. Recently, a transformer based non-autoregressive speech recognition model was proposed [10]. These models use a mask-predict manner, i.e., several tokens are replaced by the mask token randomly. And during inference, the token sequence is generated by filling the masked tokens iteratively. This method uses the predicted tokens as the language context. However, it still requires multi-pass forward propagation of the decoder to complete all the masked tokens.

We believe that the language semantic[1] is contained in the speech signal implicitly. So, if this semantic can be extracted well, the token sequence can be generated without relying on the explicit language modeling, e.g., autoregresive language models and masked language models. In this paper, we propose a *simple and effective* non-autoregressive model called LASO (Listen Attentively, and Spell Once[2]). We use the feedforward self attention mechanism [4] as basic blocks to build three modules of LASO: the encoder, the position dependent summarizer (PDS), and the decoder. The encoder encodes the acoustic features into high-level representations. The PDS summarizes the semantic at each position from the high-level representations and bridges length gap between speech and token sequence. The decoder captures token-level semantic and predicts tokens. We conduct experiments on a publicly available Chinese dataset AISHELL-1 [11]. The proposed LASO achieves 6.4% of character error rates on test set, which is better than chain model [12] and state-of-the-art autoregressive transformer models [13]. And compared with the strong baseline autoregressive transformer model, the inference of LASO speeds up by $50\times$.

2. Background

Speech recognition aims to convert an acoustic feature sequence to the corresponding textual token (word, sub-word, or phone) sequence. Given a speech-text pair (x, y), where x denotes the acoustic feature sequence, and y denotes the token sequence, the autoregressive Seq2Seq model estimates the conditional probability $P(y|x)$ by decomposition with the chain rule:

$$P(y|x) = P(y_1|x) \prod_{i=2}^{L} P(y_i|y_{<i}, x), \tag{1}$$

where y_i denotes the token at step i, $y_{<i}$ denotes the subsequence $[y_1, \cdots, y_{i-1}]$, and L denotes the length of token sequence. For an autoregressive model, the prediction of one token relies on the previously predicted tokens at inference stage.

The non-autoregressive Seq2Seq models assume that each token is independent from the others:

$$P(y|x) = \prod_{i=1}^{L} P(y_i|x). \tag{2}$$

Because the prediction does not depend on other tokens, the non-autoregressive Seq2Seq model can predict the token at each step in parallel.

[1]In this paper, we refer to the relationship among tokens as language semantic.

[2]This name is inspired by [3].

http://dx.doi.org/10.21437/Interspeech.2020-1600

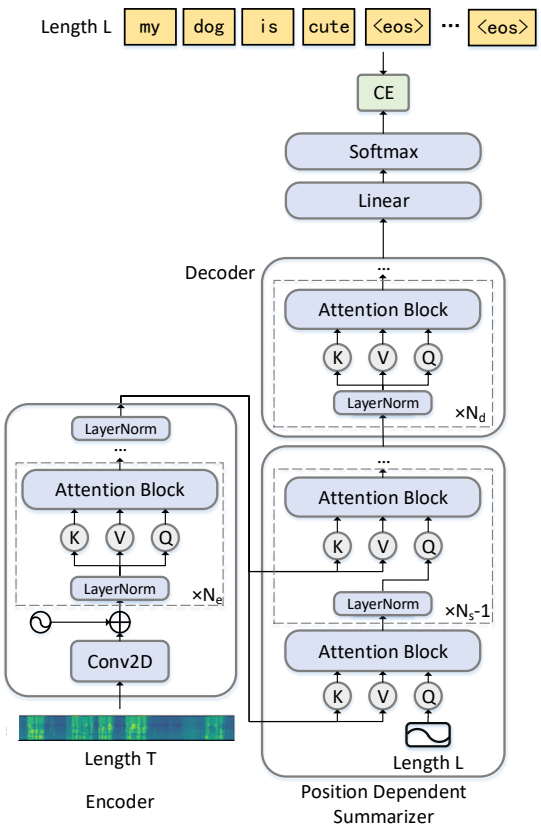

Figure 1: *The architecture of LASO. The encoder first uses a two-layer CNN to subsample the acoustic feature sequence, and then uses a stack of attention blocks to obtain high-level representations. The position dependent summarizer (PDS) queries the high-level representations for each position. It bridges the length gap between the speech sequence and the token sequence. The decoder further refines the queried outputs of the PDS. With a linear transformation, the softmax function gives the probability distribution on vocabulary at each position. The tail of the token sequence is filled by "<eos>" token. During inference, LASO directly selects the most likely token, and removes "<eos>". The network is trained with cross entropy.*

Token relationship is important for sequence generation. For the CTC based models [14], the token relationship is usually modeled by an external language model to improve performance. For RNN-Transducers [15], the token relationship is modeled with a prediction network. And for attention based encoder-decoder models, the token relationship is modeled with the decoder autoregressively. The main challenge of the non-autoregressive Seq2Seq model is: can a model generate token sequence without the explicit token relationship? We believe that the token relationship is contained in the speech implicitly. If we achieve token-level representations from speech, we can generate the token sequence with the non-autoregressive model.

3. The LASO Model

The basic idea of LASO is that the acoustic feature sequence contains not only features for pronunciation but also language semantic, i.e., relationship among tokens. If we extract representations from the whole acoustic feature sequence for each token position, we can do position-wise token prediction. Because the prediction relies on the acoustic feature sequence rather than other tokens, it can be implemented in parallel.

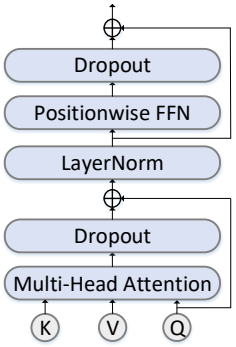

Figure 2: *An illustration of an attention block.*

Based on this idea, we formulate the position-wise token prediction as

$$\begin{aligned} z &= \text{Encode}(x), \\ P(y_i|x) &= \text{SummarizeAndDecode}(z), i = 1, 2, \cdots, L \end{aligned} \tag{3}$$

where $z = [z_1, \cdots, z_T]$ is the hidden representation sequence which has the same length with the subsampled acoustic feature sequence x. To predict the token sequence with length L, z is summarized and decoded for each position in the sequence. To generate the token sequence, the most likely token at each position is selected. The token "<eos>" is added into the vocabulary as a filler for padding the token sequence to length L. Ideally, the tail of the generated token sequence are all "<eos>", and they are easily removed after inference.

The proposed LASO consists of three modules. Each module consists of several attention blocks. The encoder encodes the acoustic feature sequence into high-level representations. The PDS summarizes the high-level representations to token-level representations based on the sinusoidal position encodings. The decoder generates the token for each position. The structure of the model is shown in Fig. 1. We first introduce the attention block. Then, we introduce each module of the model.

3.1. Attention Block

Attention mechanism has been used to model global dependency in a sequence successfully [4, 16]. Different from recurrent neural networks which represent context step-by-step, attention mechanism fuses all representations in a sequence by weighting sum. So, it can be computed in parallel. The dot-product self-attention is denoted as

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{D_k}})V, \tag{4}$$

where $Q \in \mathbb{R}^{T_q \times D_k}$, $K \in \mathbb{R}^{T_k \times D_k}$, and $V \in \mathbb{R}^{T_k \times D_v}$ denote queries, keys, and values, respectively, T_q is query sequence length, T_k is key sequence length, and D_v is the dimensionality of the keys. In this paper, D_v equals to D_k, which is set to D_m representing the model dimensionality. It can be extended to multi-head version, i.e., the hidden representations are projected into different subspaces for attention, and are concatenated together after attention [4]:

$$\begin{aligned} \text{MHA}(Q, K, V) &= \text{Concat}(h_1, \cdots, h_H)W^o, \\ h_i &= \text{Attention}(QW_i^q, KW_i^k, VW_i^v), i = 1, \cdots, H. \end{aligned} \tag{5}$$

where H is the number of heads, W^o, W_i^q, W_i^k, and W_i^v are parameter matrices. The dimensionality does not change after the multi-head attention.

The position-wise feedforward network is after the attention:

$$\text{FFN}(x) = W_2\text{Activation}(W_1 x + b_1) + b_2, \quad (6)$$

where x is a vector at one position, W_1, W_2, b_1, and b_2 are learnable parameters, Activation is a nonlinear activation function. In this work, we use gated linear units (GLUs) [17]. Residual connection [18] and layer normalization [19] are used in the attention block. We use pre-norm mechanism for stable training [20]. The attention block is the basic component of LASO.

3.2. Encoder

The first part of the encoder consists of a two layers of convolutional neural network (CNN) for capturing locality of in the feature sequence. The stride of each CNN layer is 2, so it also subsamples frame rates and compress the length of the sequence to $1/4$. Following [4], we add sinusoidal position for self attention mechanism to capture the order. Then, N_e attention blocks are used for capturing long-term relationship. Keys, queries, and values are all the inputs, i.e., self attention.

3.3. Position Dependent Summarizer

The main gap between the acoustic feature sequence and the textual token sequence is the length. Specifically, a textual token is a highly compressed semantic representation, and multiple acoustic feature frames correspond one textual token. To address this, we propose a PDS module to summarize the representations from the encoder, and to re-organize them in terms of the token positions. Basically, it is also composed of a stack of attention blocks, but the keys and the values are the outputs of the encoder. The queries of the first block are position encodings with maximum length L, and the queries of the follow-up blocks are the outputs of the previous block, as shown in Fig. 1. We use sinusoidal functions [4] to encode positions:

$$\begin{aligned} \text{pe}_{i,2j} &= \sin(i/10000^{2j/D_m}), \\ \text{pe}_{i,2j+1} &= \cos(i/10000^{2j/D_m}), \end{aligned} \quad (7)$$

where $i = 1, \cdots, L$ denotes the i-th position, and $2j$ and $2j+1$ denote element indexes in a vector. The sinusoidal position encodings provide position dependent information to query representation corresponding to specific position in token sequence from the encoder outputs. So, the sequence length matches the token sequence, i.e., the length of the outputs of PDS is L. L can be set by counting the lengths in the training set.

3.4. Decoder

After the PDS, we use the decoder to further capture token relationship. The outputs of the PDS can be seen as the representations corresponding to the tokens. So, we use self attention mechanism to capture the semantic relationship in the sequence. The decoder leverages a stack of attention blocks, and the keys, values and queries are the outputs of the previous block. After the decoder, we use a linear transformation to project the self attention based semantic representation, and softmax functions to compute probability distributions on the token vocabulary for each position.

3.5. Learning

For optimizing the parameters of the model, we minimize the position-wise cross entropy loss

$$\text{CE}(\theta) = -\frac{1}{NL} \sum_{(x,y)\in\mathcal{D}} \sum_{i=1}^{L} \log P(y_i|x;\theta). \quad (8)$$

Table 1: *The description of AISHELL-1. "Length" means the average number of tokens per utterance.*

	#Utter.	#Hour	Length	#Speaker
Training	120,098	150	14.4	340
Development	14,326	18	14.3	40
Test	7,176	10	14.6	20

where $\mathcal{D}$ is the dataset which contains N pairs of speech and token sequence (x, y), L is the maximum length we pad to, and y_i is the token at position i in token sequence y.

3.6. Inference

For decoding, we just select the token which has the highest probability at each position. Given an acoustic feature sequence, the predicted token at position i is

$$\hat{y_i} = \arg\max_{y_i} P(y_i|x;\theta). \quad i = 1, \cdots, L, \quad (9)$$

After prediction, the filler tokens "<eos>" at the tail of the sequence are removed.

4. Experiments

4.1. Datasets

We conduct experiments on a publicly available Chinese Mandarin corpus AISHELL-1 [11]. The dataset includes about 150 hours of speech for training, about 18 hours of speech for development, and about 10 hours speech for test. The speakers of training set, development set, and test set are not overlapped. All the recordings are in 16 kHz WAV format.

4.2. Experimental Setup

We use 80-dimension Mel-filter bank features (FBANK) as the input, which are extracted every 10ms with 25ms of frame length. The token vocabulary contains 4231 characters in training set and two special symbols, i.e., "<unk>" for unseen characters and "<eos>" as the filler of the tail of a token sequence.

The structure of the LASO model is shown in Fig. 1. Each layer of the two-layer subsampling CNN consists of 32 convolution filters with size 3×3, and the stride on time axis is 2. The activation functions of the CNN are ReLUs. All the attention blocks used in the model are the same. Both the encoder and the decoder have 6 attention blocks, i.e., $N_e = N_d = 6$ in Fig. 1. All the attention blocks have 8 heads. We compare different numbers of the attention blocks of PDS, i.e., $N_s = 1, 2, 3$ and 4, respectively. The intermediate dimensionality of the position-wise feedforward network is 2048, and the activation function is GLU. We train two types of LASO with different model dimensionalities D_m. We refer to the model with $D_m = 512$ as LASO-base, and the model with $D_m = 768$ as LASO-big.

We re-implement Speech-Transformer as the baseline [5]. It uses the same CNN as our LASO architecture. Following their configuration, both encoder and the decoder have 6 layers. The dimensionality of the model is 512, and the intermediate dimensionality of the position-wise feedforward network is 2048. The number of heads of the multi-head attention is 8.

All models are trained with the same procedure. We use the Adam algorithm to train the model for 130 epochs. Each batch contains about 100 seconds of speech, and we accumulate gradients of 12 steps for simulating big batch [21]. We follow the warm-up learning rate schedule [4]:

$$\alpha = D_{\text{m}}^{-0.5} \cdot \min(step^{-0.5}, step \cdot warmup^{-1.5}), \quad (10)$$

and the warm-up step is set to 12000. To avoiding overfitting, we set dropout rate to 0.1. We use SpecAugment [22] for data

Table 2: *Character Error Rates (CERs) on the development set of the models with different numbers of attention blocks of PDS.*

#block of PDS	1	2	3	4
LASO-base	6.4	6.5	6.5	6.4
LASO-big	6.2	6.2	6.3	6.2

Table 3: *The character error rates (CERs) of the systems on AISHELL-1. Latency is inference time per utterance on test set (including time of feature extraction). Real-time factor (RTF) is computed as the ratio of the total inference time to the total duration of the test set. Inference is done utterance by utterance without batching, on an NVIDIA RTX 2080Ti GPU.*

System	CER %		RTF/Latency
	Dev.	Test	
KALDI (nnet3) * † ‡	-	8.6	-
KALDI (chain) * † ‡	-	7.4	-
ESPNet (Transformer) † [13]	6.0	6.7	-
A-FMLM [10]	6.2	6.7	-
Fan et al. [24]	-	6.7	-
Transformer (ours)	6.1	6.6	0.19 / 961ms
LASO-base	6.4	7.3	0.0034 / 17ms
LASO-base + speed perturb	6.0	6.8	0.0034 / 17ms
LASO-big	6.2	7.0	0.0043 / 21ms
LASO-big + speed perturb	**5.8**	**6.4**	0.0043 / 21ms

* from the KALDI official repository.
† with speed perturbation based data augmentation.
‡ with i-vector based speaker adaptation.

augmentation, and we leverage label smoothing with 0.1 during training. We average parameters of the models which are saved at the last 10 epochs as the final model. The maximum length L is set to 60, which is set by counting the characters in the utterances of the training set.

All the systems are implemented with PyTorch [23].All the experiments are conducted on an NVIDIA RTX 2080Ti GPU. For evaluating inference speed, we predict one utterance once for evaluate speed, i.e., the batch contains 1 utterance. For the autoregressive models, beam-width is set to 5 and the maximum decoding length is 60.

4.3. Results

We first compare the LASO with different numbers of attention blocks of PDS on the development set. Table 2 shows the results. We can see that different numbers of attention blocks of PDS impact the performance, but the difference is small. In the rest of the experiments, we use 4 attention blocks in the PDS module, i.e., $N_s = 4$.

The performances are shown in Table 3. We can see that the LASO models achieve good performance with very low latency. LASO-base achieves a 7.3% of CER on the test set. LASO-big achieves a 7.0% of CER on the test set. Both LASO-base and LASO-big outperform chain model (7.4%) [12], without speed perturbation. And it is very close to the state-of-the-art transformers (6.7%) and our re-implemented transformer model (6.6%). These results confirm our idea: if the implicit language semantic is captured, prediction of tokens without explicit relationship among tokens is feasible.

The performance of the bigger model LASO-big is better than LASO-base. The large scale parameters and more layers make the model more powerful to extract token-level semantic representation for each position. To further improve the performance, we augment training data with speed perturbation [25], and retrain the two LASO models. We use factors 0.9 and 1.1 to perturb the speed of the audio and combine the augmented data with the original data. With speed perturbation, the CERs of LASO-base and LASO-big are further reduced to 6.8% and 6.4%, respectively.

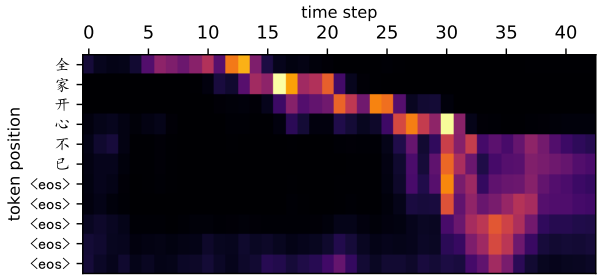

Figure 3: *Visualization of the attention scores of the 4-th attention block. The horizontal axis is the time step of the outputs of the encoder, and the vertical axis is the token sequence. The token sequence mean "the family are very happy". We only show the first 11 token positions for saving space.*

We also show inference speed in Table 3. We can see that the latency of LASO models is much smaller than autoregressive models. The speed-up is about 50×. The non-autoregressive structure makes LASO do not need multi-pass forward computation in beam-search. And the feed-forward structure of LASO makes parallel computation efficient.

To better understand the behaviors of the PDS module, we visualize the attention scores of the 4-th attention blocks of the PDS in LASO-big. The attention scores are the average of the 8 heads. We can see that the alignments of the meaningful tokens and the outputs of the encoder are from the upper-left to the bottom-right. For the filler token <eos>, the alignment is vague. Because no certain correspondence between the filler token and the outputs of the encoder exists. Because different head has different alignment in the multi-head attention, the averaged scores are not very sharp.

5. Conclusions and Future Works

In this paper, we propose a new non-autoregressive speech recognition model. We assume that speech signal contains the relationship among tokens implicitly, and token sequence can be generated without explicit language modeling. Based on this, we propose the LASO model. LASO forward propagates only one-pass for token generation, without beam-search. And because of the feedforward structure, parallel computation can be implemented efficiently, and time cost of inference can be significantly reduced. Experiments demonstrate that the proposed models have very low latency and promising performances. This work is the first result of the LASO model. In the future, we will investigate how to improve the performance of the LASO model by loss functions.

6. Acknowledgements

This work is supported by the National Key Research & Development Plan of China (No.2016YFB1001404), the National Natural Science Foundation of China (NSFC) (No.61831022, No.61901473, No.61771472, No.61773379) and Inria-CAS Joint Research Project (No.173211KYSB20170061 and No.173211KYSB20190049). This work is also (partially) funded by Huawei Noah's Ark Lab. We also thank the anonymous reviewers for their invaluable comments.

7. References

[1] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," *international conference on acoustics, speech, and signal processing*, pp. 4945–4949, 2016.

[2] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 4835–4839.

[3] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[5] L. Dong, S. Xu, and B. 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)*. IEEE, 2018, pp. 5884–5888.

[6] J. Gu, J. Bradbury, C. Xiong, V. O. Li, and R. Socher, "Non-autoregressive neural machine translation," *arXiv preprint arXiv:1711.02281*, 2017.

[7] J. Lee, E. Mansimov, and K. Cho, "Deterministic non-autoregressive neural sequence modeling by iterative refinement," *arXiv preprint arXiv:1802.06901*, 2018.

[8] M. Ghazvininejad, O. Levy, Y. Liu, and L. Zettlemoyer, "Mask-predict: Parallel decoding of conditional masked language models," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 6114–6123.

[9] X. Ma, C. Zhou, X. Li, G. Neubig, and E. Hovy, "Flowseq: Non-autoregressive conditional sequence generation with generative flow," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 4273–4283.

[10] N. Chen, S. Watanabe, J. Villalba, and N. Dehak, "Listen and fill in the missing letters: Non-autoregressive transformer for speech recognition," *arXiv preprint arXiv:1911.04908*, 2019.

[11] 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 *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. IEEE, 2017, pp. 1–5.

[12] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi." in *Interspeech*, 2016, pp. 2751–2755.

[13] 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," *arXiv preprint arXiv:1909.06317*, 2019.

[14] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[15] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[16] C. Raffel and D. P. Ellis, "Feed-forward networks with attention can solve some long-term memory problems," *arXiv preprint arXiv:1512.08756*, 2015.

[17] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 933–941.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[19] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[20] T. Q. Nguyen and J. Salazar, "Transformers without tears: Improving the normalization of self-attention," *arXiv preprint arXiv:1910.05895*, 2019.

[21] M. Ott, S. Edunov, D. Grangier, and M. Auli, "Scaling neural machine translation," in *Proceedings of the Third Conference on Machine Translation: Research Papers*, 2018, pp. 1–9.

[22] 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," *arXiv preprint arXiv:1904.08779*, 2019.

[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.

[24] Z. Fan, S. Zhou, and B. Xu, "Unsupervised pre-traing for sequence to sequence speech recognition," *arXiv preprint arXiv:1910.12418*, 2019.

[25] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition." pp. 3586–3589, 2015.