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

Align-Denoise: Single-Pass Non-Autoregressive Speech Recognition

Nanxin Chen[1], *Piotr Żelasko*[1,2], *Laureano Moro-Velázquez*[1], *Jesús Villalba*[1,2], *Najim Dehak*[1,2]

[1]Center for Language and Speech Processing, Johns Hopkins University, USA
[2]Human Language Technology Center of Excellence, Johns Hopkins University, USA
{nchen14, pzelasko, laureano, jvillal7, ndehak3}@jhu.edu

Abstract

Deep autoregressive models start to become comparable or superior to the conventional systems for automatic speech recognition. However, for the inference computation, they still suffer from inference speed issue due to their token-by-token decoding characteristic. Non-autoregressive models greatly improve decoding speed by supporting decoding within a constant number of iterations. For example, Align-Refine was proposed to improve the performance of the non-autoregressive system by refining the alignment iteratively. In this work, we propose a new perspective to connect Align-Refine and denoising autoencoder. We introduce a novel noisy distribution to sample the alignment directly instead of obtaining it from the decoder output. The experimental results reveal that the proposed Align-Denoise speeds up both training and inference with performance improvement up to 5% relatively using single-pass decoding.

Index Terms: speech recognition, non-autoregressive model, deep learning, denoising autoencoder, iterative refinement

1. Introduction

Sequence-to-sequence models have become increasingly popular nowadays in the speech research community [1–9]. They are introduced to solve the sequence prediction problem as a specific type of end-to-end model. Sequence-to-sequence models take the input sequence and predict the target sequence directly, without requiring multiple modules separately optimized. However, most common sequence-to-sequence models usually predict one label at a time. These are widely known as *autoregressive models*.

Autoregressive models prediction at a given time step relies on the predictions or ground truth values of tokens at previous time steps. Autoregressive models are simple to train and are theoretically well motivated by linking them to the probability chain rule. Unfortunately, their simplicity results in inefficiency during inference, particularly when dealing with long output sequences. Considering the fact that output tokens need to be predicted one-by-one and that each prediction requires a full forward pass through the decoder, the computation time depends heavily on the output sequence length, thus the inference procedure is very slow.

The need to speed up the inference procedure results in a growing interest in a different type of sequence-to-sequence model, known as *non-autoregressive models* [10–20]. In contrast to autoregressive models, non-autoregressive models predict the whole output sequence within a constant number of iterations, independently of the output sequence length. This leads to a large speedup of decoding, especially for long output utterances.

Audio-Conditional Masked Language Model (A-CMLM) and Audio-Factorized Masked Language Model (A-FMLM) [20] are the first attempts to introduce non-autoregressive models for speech recognition, which predict the text sequence from the input recordings. In these approaches, tokens are randomly masked during training, and the network is optimized to recover their ground-truth values. An extension to these methods includes using Connectionist Temporal Classification (CTC) greedy decoding result to initialize the hypothesis [11], applying a similar approach on the frame-level CTC alignments [10].

Another direction for the non-autoregressive ASR is based on iterative refinement [21], which predicts the whole sequence within each iteration. Iterative refinement reduces train-test mismatch from the masking-based approaches. Align-Refine [16] optimizes the decoder to refine the CTC latent alignments from the greedy decoding. The advantage of working at the alignment level is that the output sequence length doesn't need to be decided in the beginning. Besides, it is easier to fix some insertion or deletion errors at the alignment level by changing predictions of few frames.

The alignments used for Align-Refine training come from two different sources. The first source is the *initial proposal* generated by the encoder–the same that is conventionally used for the CTC objective in joint attention and CTC training [22]. The second source includes predictions from the decoder. To match the inference, Align-Refine requires several passes through the decoder to get the intermediate alignments to train the network. By obtaining those results, Align-Refine benefits from the iterative decoding during the inference. However, that approach results in a drawback of increased computation time and memory consumption.

To address these issues, we propose *Align-Denoise*, which doesn't require multiple decoder passes during training. Instead of collecting the intermediate results by forward-passing, we sample them from a proposed noisy distribution of alignment. The noisy distribution mixes the result of the initial proposal and the ground-truth CTC alignment. The network is optimized to refine the sampled alignment from the noisy distribution. By incorporating the idea of denoising autoencoder [23], Align-Denoise not only avoids multiple forward passes during training but also obtains better results with single-pass decoding during inference.

The main contributions of this paper are as follows:

- A novel perspective to extend Align-Refine inspired by denoising autoencoder.
- A noise distribution to sample alignment based on the initial proposal and ground-truth alignment.
- A new framework, Align-Denoise, which uses sampled alignment to train and outperforms Align-Refine with speedup on both training and inference.
- We will release our code by publication time.

http://dx.doi.org/10.21437/Interspeech.2021-1906

This paper is organized as follows. Section 2 introduces the Align-Refine framework, which uses iterative refinement to generate the transcript. Inspired by the denoising autoencoder, Align-Denoise is proposed to speed up the training, which is described with details in Section 3. Section 4 introduces the experimental setup used for training and evaluation. Discussions are given in Section 5. Section 6 summarizes the conclusions and proposes directions for future research.

2. Align-Refine

The Align-Refine model [16] can be considered as an extension to the Connectionist Temporal Classification (CTC) [24]. CTC simplifies the speech recognition decoding by assuming conditional independence between predictions,

$$p(\mathbf{a} \mid \mathbf{x}) = \prod_{t=0}^{T-1} p(\mathbf{a}_t \mid \mathbf{x}) \tag{1}$$

where T is the number of frames, $\mathbf{x}$ are the features of the input audio, $\mathbf{a}$ is the alignment that can be mapped to the final result $\mathbf{y}$. To measure the likelihood efficiently and exactly, CTC uses dynamic programming to marginalize over all possible alignments. However, this independence assumption is too strong for speech recognition since it ignores the dependence between tokens at different positions. Consequently, dependencies related to the language model cannot be easily learned by CTC training.

To alleviate the conditional independence assumption, a common solution is to provide an additional conditional signal which provides *a priori* alignment information:

$$p(\mathbf{a} \mid \mathbf{x}, \hat{\mathbf{a}}) = \prod_{t=0}^{T-1} p(\mathbf{a}_t \mid \mathbf{x}, \hat{\mathbf{a}}) \tag{2}$$

where $\hat{\mathbf{a}}$ contains alignment related information. Previous works explored different forms of $\hat{\mathbf{a}}$. Imputer [10] uses a partial result as the conditional signal, and the unfinished positions are labeled by the special token MASK. In comparison, Align-Refine and this work condition the likelihood on noisy alignment $\hat{\mathbf{a}}$ which may provide more information than the partial result.

Align-Refine optimizes a non-causal decoder to refine the initial decoding result from the encoder. Instead of training separate models for each step, Align-Refine applies the same decoder iteratively on the previous predicted alignment, starting from the encoder's prediction. During training, this recursive refinement has been unfolded for K iterations but there are no gradient flows between different iterations. Thus Align-Refine requires K forward passes through the decoder but during backpropagation, they are treated independently.

Align-Refine can be viewed as a special denoising autoencoder [23]. The goal of the denoising autoencoder is to learn the mapping from the noisy sample to the clean one:

$$\mathbb{E}_{\mathbf{a}\sim p(\mathbf{a})}\, \mathbb{E}_{\tilde{\mathbf{a}}\sim q(\tilde{\mathbf{a}}|\mathbf{a})} \left[\left\| f_{\mathsf{dec}}(\mathbf{a} \mid \mathbf{x}, \tilde{\mathbf{a}}) - \mathbf{a} \right\|_2^2 \right] \tag{3}$$

to optimize the network f_{dec} where $q(\tilde{\mathbf{a}} \mid \mathbf{a})$ is the noise distribution. For the case of ASR, additional conditional signal x is available and the CTC objective is used instead:

$$\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\, \mathbb{E}_{\tilde{\mathbf{a}}\sim q(\tilde{\mathbf{a}}|\mathbf{x},\mathbf{a})}\, L_{\mathsf{CTC}}\left[f_{\mathsf{dec}}(\mathbf{a} \mid \mathbf{x}, \tilde{\mathbf{a}}), \mathbf{y} \right]. \tag{4}$$

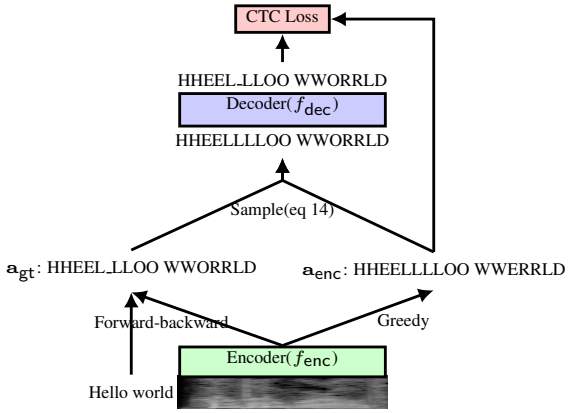

Figure 1: *A visualization of the Align-Denoise training. Two signals are estimated from the encoder output: the initial proposal* $\mathbf{a}_{\mathsf{enc}}$ *from greedy decoding and the ground-truth alignment* $\mathbf{a}_{\mathsf{gt}}$ *from the forward-backward algorithm. The noisy transcript is sampled by mixing these two sequences. The decoder, a non-causal Transformer decoder, conditions on this and the encoder to improve the alignment. The connection between encoder and decoder is omitted.*

The noise distribution in Align-Refine has been simplified to the form $q(\tilde{\mathbf{a}} \mid \mathbf{x})$ and is a collection of sequences $\{f_{\mathsf{enc}}(\mathbf{x}), f_{\mathsf{dec}}(f_{\mathsf{enc}}(\mathbf{x})), ..., f^K_{\mathsf{dec}}(f_{\mathsf{enc}}(\mathbf{x}))\}$ where $f_{\mathsf{enc}}(\mathbf{x})$ is the decoding result from the encoder, $f^i_{\mathsf{dec}}(f_{\mathsf{enc}}(\mathbf{x}))$ is the result after applying the decoder i times.

3. Align-Denoise

The noise distribution $q(\tilde{\mathbf{a}} \mid \mathbf{a}, \mathbf{x})$ used in Align-Refine requires K forward passes through the decoder. This increases the training time by a large margin, and it is impossible to increase the number of iterations K to an arbitrary number since it is limited by the GPU memory. Another important factor is that the ground-truth alignment $\mathbf{a}$ is not used in the noise distribution. In this case, we may not have enough training samples to cover the space near $\mathbf{a}$ if the initial proposal is far away from $\mathbf{a}$.

Align-Denoise speeds up the training process of Align-Refine by incorporating the ground-truth alignment $\mathbf{a}$. Instead of taking a fixed number of steps from the initial proposal generated by the encoder, Align-Denoise samples alignments that combine the ground-truth alignment and the initial proposal.

The initial proposal $\mathbf{a}_{\mathsf{enc}}$ is generated by the encoder using greedy decoding,

$$\mathbf{a}_{\mathsf{enc}} = \arg\max f_{\mathsf{enc}}(\mathbf{a} \mid \mathbf{x}) \tag{5}$$

which can be estimated efficiently by taking argmax separately at each position due to the conditional independence assumption.

The posterior alignment provided by the encoder given the ground truth transcription $\mathbf{y}$ is

$$P(\mathbf{a}_t = k \mid \mathbf{x}, \mathbf{y}, f_{\mathsf{enc}}) = \sum_{\mathbf{a}:\phi(\mathbf{a})=\mathbf{y}\ \mathbf{a}_t=k} f_{\mathsf{enc}}(\mathbf{a} \mid \mathbf{x}) \tag{6}$$

where ϕ is the mapping function from CTC alignment to the label sequence. The posterior can be estimated by a dynamic programming algorithm known as the forward-backward algorithm [25]. This posterior is the probability of $\mathbf{a}_t = k$ among

all acceptable alignments and the condition of acceptable alignments is $\phi(\mathbf{a}) = \mathbf{y}$. Alternatively, it can be viewed as a weighted finite-state acceptor (WFSA) that represents all possible alignments with frame-level label likelihoods.

The ground-truth alignment can be derived from the posterior by

$$\mathbf{a}_{\text{gt}} = g(\mathbf{x}, \mathbf{y}, f_{\text{enc}}) = \arg\max_k P(\mathbf{a}_t = k \mid \mathbf{x}, \mathbf{y}, f_{\text{enc}}) \quad (7)$$

During training, noisy alignments $\tilde{\mathbf{a}}$ are sampled from the special noise distribution following our alignment policy. Similar to Align-Refine, the label posterior is computed by marginalizing over all possible alignments $\mathbf{a}$ which correspond to the target label sequence,

$$P(\mathbf{y}|\mathbf{x}, \tilde{\mathbf{a}}, f_{\text{dec}}) = \sum_{\mathbf{a}:\phi(\mathbf{a})=\mathbf{y}} f_{\text{dec}}(\mathbf{a} \mid \mathbf{x}, \tilde{\mathbf{a}}) \quad (8)$$

$$= \sum_{\mathbf{a}:\phi(\mathbf{a})=\mathbf{y}} \prod_{t=0}^{T-1} f_{\text{dec}}(\mathbf{a}_t \mid \mathbf{x}, \tilde{\mathbf{a}}) \quad (9)$$

The factorization of $\prod_{t=0}^{T-1} f_{\text{dec}}(\mathbf{a}_t \mid .)$ is due to the conditional independence assumption, which results in non-autoregressive decoding.

Finally, the encoder and decoder are jointly trained by minimizing this modified CTC loss,

$$L_{\text{CTC}} = -\mathbb{E}_{\mathbf{x},\mathbf{y}\sim p(\mathbf{x},\mathbf{y})} \mathbb{E}_{\mathbf{a}_{\text{gt}}=g(\mathbf{x},\mathbf{y},f_{\text{enc}})} \mathbb{E}_{\tilde{\mathbf{a}}\sim q(\tilde{\mathbf{a}}|\mathbf{x},\mathbf{a}_{\text{gt}})} \left[\log P(\mathbf{y}|\mathbf{x}, \tilde{\mathbf{a}}, f_{\text{dec}})\right] \quad (10)$$

where $p(\mathbf{x}, \mathbf{y})$ is the data distribution, $\mathbf{a}_{\text{gt}}$ are the ground truth alignments obtained by (7); and $q(\tilde{\mathbf{a}} \mid \mathbf{a}_{\text{gt}}, \mathbf{x})$ is the noisy distribution that generates noisy alignments given the ground truth alignments.

3.1. Noise Distribution

The key idea behind Align-Denoise is to design an alignment policy that can generate noisy transcripts, similar to the intermediate results encountered during inference. Given that the input sequence is discrete, we add noise to the posterior instead and sample from it.

If the greedy decoding $\mathbf{a}_{\text{enc}}$, obtained from the encoder in (5), is adopted as the initialization, our goal is to find a mapping from $\mathbf{a}_{\text{enc}}$ to the true alignment $\mathbf{a}_{\text{gt}}$. This is achieved by adding the noise to the encoder posterior given ground truth labels,

$$P_{\text{gt}}(\mathbf{a}_t = k) = P(\mathbf{a}_t = k \mid \mathbf{x}, \mathbf{y}, f_{\text{enc}}) \quad (11)$$

given by (6). From now on, we will denote $P_{\text{gt}}(\mathbf{a}_t = k)$ as *ground truth posterior*. The errors made by the encoder correspond to the set of indices $r \in R$ such that

$$\mathbf{a}_{\text{enc},r} \neq \mathbf{a}_{\text{gt},r} \quad (12)$$

To sample the noisy alignment $\tilde{\mathbf{a}}$, we keep the correct predictions from the greedy encoder result consistent and sample the rest frames:

$$V_t(k) \sim \mathcal{N}(\sqrt{\alpha} * P_{\text{gt}}(\mathbf{a}_t = k), (1-\alpha) * \sigma_t^2(k) * \mathbb{1}(t \in R)) \quad (13)$$

$$\tilde{\mathbf{a}}_t = \arg\max_k V_t \quad (14)$$

where α is sampled from $[0, 1]$ to control the global noise level like [26], V_t is the sampled per-frame distribution of labels. Intuitively, we choose frames based on the correctness of the initial proposal and sample frame-level labels based on combined uncertainties σ_t from CTC decoding and forward-backward algorithm. For a specific position, the amount of noise in sampling comes directly from model's uncertainty.

To mimic the error made by the encoder greedy decoding, the noise variance is defined as the combination of both greedy result and true posterior,

$$\sigma_t^2(k) = \max(P_{\text{gt}}(\mathbf{a}_t = k), \lambda \times P_{\text{enc}}(\mathbf{a}_t = k)), \quad (15)$$

with

$$P_{\text{enc}}(\mathbf{a}_t) = f_{\text{enc}}(\mathbf{a}_t \mid \mathbf{x}). \quad (16)$$

From now on, we will denote $P_{\text{enc}}(\mathbf{a}_t)$ as the *encoder posterior*. λ controls the weight of the greedy decoding result. When λ is large, the input is mainly determined by the encoder result so that it may contain more errors. When λ is small, the input is close to the ground-truth alignment which includes fewer mistakes.

Figure 1 includes visualization of the proposed Align-Denoise training process. CTC losses are measured on both initial proposal and decoder prediction following other previous work [16, 22]. Table 1 demonstrates how our policy generates different training samples. The encoder greedy result makes several mistakes, for example, the words 'Wang' and 'tell'. The training examples keep some of them, and the model is optimized to correct.

4. Experiment

To evaluate the effectiveness of the proposed model, we conduct speech recognition experiments to compare different encoder-decoder models. The performance of the models is evaluated based on word error rates (WERs) without relying on external language models and beam search.

All models are tested on the 81-hours Wall Street Journal (WSJ) [27]. For the network inputs, we use 80 Mel-scale filterbank coefficients with three-dimensional pitch features and apply SpecAugment [28] during model training. We run all models at the character level to match previous works [10, 16].

4.1. Network Architecture

For all experiments, we adapt the same encoder-decoder architecture as Mask CTC [29] and Align-Refine [16], which consists of Transformer self-attention layers with 4 attention heads, 256 hidden units, and 2048 feed-forward inner dimension size. The encoder includes 12 self-attention layers with convolutional layers that downsample the input features by a factor of 4. The decoder contains 6 non-causal self-attention layers.

We set the dropout rate to 0.1 and the weight of the initial proposal to 0.3. Each minibatch includes 48 sequences, and gradients are accumulated from 8 batches. The model is trained for 500 epochs following Mask-CTC [11] with a standard inverse square-root learning rate schedule and a linear warmup of 25000 steps.

4.2. Results

Table 2 shows the results for WSJ based on WERs and real-time factors (RTFs) that were measured for decoding the eval92 subset on a single CPU thread. By comparing the results for non-autoregressive models, we can see that the proposed method

Table 1: *Training samples generated by our policy. The noise distribution is a Gaussian distribution whose variance is controlled by the confidence from the encoder initial proposal (enc) and posterior from forward-backward algorithm. Training samples inherit some predictions errors from the initial proposal.*

posterior	_____SSOO_ MR.. WANNNG TEL__LS PEOPLLE HEE IIS FFIFFTYYY_____
enc	_____SSOO_ MR.. WIGNNNTTELLLLS PEOPLLE HE IIS FFIFFTYYY_____
sample 1	_____SSOO_ MR.. WINNNG TELL_LS PEOPLLE HE IIS FFIFFTYYY_____
sample 2	_____SSOO_ MR.. WANNNT_TELL_LS PEOPLLE HE' IIS FFIFFTYYY_____
sample 3	_____SSOO_ MR.. WAHNNGTTEL_ELS PEOPLLE HE IIS FFIFFTYYY_____
sample 4	_____SSOO_ MR.. WANNNG_TELLLLS PEOPLLE HE IIS FFIFFTYYY_____
sample 5	_____SSOO_ MR.. WENNNGTTELLLLS PEOPLLE HE IIS FFIFFTYYY_____

Table 2: *Word error rates (WERs) and real time factor (RTF) for WSJ (English).*

Model	Iterations	dev93(↓)	eval92(↓)	RTF(↓)
Autoregressive				
CTC-attention [22]	L	14.4	11.3	0.97
+ beam search	L	13.5	10.9	4.62
Align-Denoise				
$\lambda = 0.5$	1	13.8	11.3	0.05
$\lambda = 0.3$	1	**13.5**	**10.8**	0.05
$\lambda = 0.1$	1	14.4	11.4	0.05
Previous work				
CTC [11]	-	22.2	17.9	0.03
CTC [10]	-	–	15.2	–
Imputer (IM) [10]	8	–	16.5	–
Imputer (DP) [10]	8	–	12.7	–
Mask CTC [11]	1	15.7	12.5	0.07
Mask CTC [11]	10	15.5	12.2	0.07
Align-Refine [16]	1	14.1	11.6	0.05
Align-Refine [16]	5	13.7	11.4	0.07

outperforms all other non-autoregressive baselines using just a single iteration. $\lambda = 0.3$ gives the best result, whereas the performance of $\lambda = 0.1$ is worse in comparison. When λ is small, the input alignment is too close to the ground truth, giving the network limited opportunities to improve it. However, during inference, the initial proposal contains more mistakes. Such mismatch can explain the reason why a small λ is not preferred, which is verified by the result.

Since we use the same architecture, the real-time factor of Align-Denoise matches Align-Refine with $k = 1$ iteration. Comparing to the Align-Refine, the proposed Align-Denoise also speeds up the training. Align-Refine requires $K = 4$ forward passes through the decoder, which consists of 6 transformer blocks. The proposed approach requires just a single forward pass through the decoder since noisy alignments are sampled from frame-level Gaussian distributions instead of using the greedy results from the forward passes.

5. Discussion

Iterative Refinement. For the proposed Align-Denoise, we observe no improvement during inference when we run it for more than 1 iteration. The model still makes changes but they do not affect the post-collapse outputs. Even for as many as fifty iterations, the performance remains the same.

Alignment mismatch. There exist some cases where $\mathbf{a}_{\text{enc}}$ and $\mathbf{a}_{\text{gt}}$ are mis-aligned. One example is given in Table 3. In the initial proposal from the encoder, the last 'S' in the word *prices* has been repeated twice – which does not change the decoding result. However, it makes the alignment mismatched and there is not enough character space to recognize the word *thing*. The only way to correct this is to remove the extra 'S' and move the whole segment to the left. Similar problems have been observed in other non-autoregressive approaches [11, 20] that predict tokens directly instead of the alignment. Reducing the downsampling rate in the encoder could mitigate this kind of error at the cost of decoding speed.

Table 3: *Mis-aligned example. This is part of the sentence and the original sentence is* ***the meteoric rise in prices is the worst thing that could happen to the collectible car hobby.***

$\mathbf{a}_{\text{enc}}$	PPRICESS ISS THE WORSTTHAN THAT COUD HAVEPENN
$\mathbf{a}_{\text{gt}}$	PPRICES IS THE WORST THING THAT COULD HAP_PEN

6. Conclusion

In this paper, we presented Align-Denoise, a non-autoregressive speech recognition model which uses single-pass decoding. Align-Denoise is built on the prior work of Align-Refine [16], which iteratively refines the CTC latent alignment. Inspired by a theoretical connection between Align-Refine and denoising autoencoder, we propose a novel noisy distribution which can sample the alignment directly. This speeds up the Align-Refinement training by reducing the number of forward passes through the decoder. Experiments demonstrate that Align-Denoise leads to a small performance improvement compared to the Align-Refine with a single iteration. Future work includes improving the performance with multiple iterations using score matching [30, 31] and adapting the noise distribution to support mis-aligned cases.

7. References

[1] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[2] 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.

[3] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N.-E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, "ESPnet: End-to-end speech processing toolkit," *Proc. Interspeech 2018*, pp. 2207–2211, 2018.

[4] 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.

[5] 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.

[6] C. Lüscher, E. Beck, K. Irie, M. Kitza, W. Michel, A. Zeyer, R. Schlüter, and H. Ney, "Rwth asr systems for librispeech: Hybrid vs attention," *Interspeech, Graz, Austria*, pp. 231–235, 2019.

[7] G. Synnaeve, Q. Xu, J. Kahn, E. Grave, T. Likhomanenko, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, "End-to-end asr: from supervised to semi-supervised learning with modern architectures," *CoRR*, vol. abs/1911.08460, 2019.

[8] P. Guo, F. Boyer, X. Chang, T. Hayashi, Y. Higuchi, H. Inaguma, N. Kamo, C. Li, D. Garcia-Romero, J. Shi *et al.*, "Recent developments on espnet toolkit boosted by conformer," *arXiv preprint arXiv:2010.13956*, 2020.

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

[10] W. Chan, C. Saharia, G. Hinton, M. Norouzi, and N. Jaitly, "Imputer: Sequence modelling via imputation and dynamic programming," *arXiv preprint arXiv:2002.08926*, 2020.

[11] Y. Higuchi, S. Watanabe, N. Chen, T. Ogawa, and T. Kobayashi, "Mask ctc: Non-autoregressive end-to-end asr with ctc and mask predict," *arXiv preprint arXiv:2005.08700*, 2020.

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

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

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

[15] Y. Higuchi, H. Inaguma, S. Watanabe, T. Ogawa, and T. Kobayashi, "Improved mask-ctc for non-autoregressive end-to-end asr," *arXiv preprint arXiv:2010.13270*, 2020.

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

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

[18] X. Song, Z. Wu, Y. Huang, C. Weng, D. Su, and H. Meng, "Non-autoregressive transformer asr with ctc-enhanced decoder input," *arXiv preprint arXiv:2010.15025*, 2020.

[19] R. Fan, W. Chu, P. Chang, and J. Xiao, "Cass-nat: Ctc alignment-based single step non-autoregressive transformer for speech recognition," *arXiv preprint arXiv:2010.14725*, 2020.

[20] 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, 2021.

[21] J. Lee, E. Mansimov, and K. Cho, "Deterministic non-autoregressive neural sequence modeling by iterative refinement," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 1173–1182.

[22] 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.

[23] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[24] 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.

[25] L. E. Baum *et al.*, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes," *Inequalities*, vol. 3, no. 1, pp. 1–8, 1972.

[26] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan, "Wavegrad: Estimating gradients for waveform generation," in *International Conference on Learning Representations*, 2021.

[27] D. B. Paul and J. M. Baker, "The design for the wall street journal-based CSR corpus," in *Proceedings of Workshop on Speech and Natural Language*, 1992.

[28] 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 *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[29] S. Karita, N. E. Y. Soplin, S. Watanabe, M. Delcroix, A. Ogawa, and T. Nakatani, "Improving Transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *Proc.Interspeech*, 2019.

[30] A. Hyvärinen and P. Dayan, "Estimation of non-normalized statistical models by score matching." *Journal of Machine Learning Research*, vol. 6, no. 4, 2005.

[31] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural computation*, vol. 23, no. 7, pp. 1661–1674, 2011.