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

Mixture Model Attention: Flexible Streaming and Non-Streaming Automatic Speech Recognition

Kartik Audhkhasi, Tongzhou Chen, Bhuvana Ramabhadran, Pedro J. Moreno

Google, USA
{kaudhkhasi, tongzhou, bhuv, pedro}@google.com

Abstract

Streaming automatic speech recognition (ASR) hypothesizes words as soon as the input audio arrives, whereas non-streaming ASR can potentially wait for the completion of the entire utterance to hypothesize words. Streaming and non-streaming ASR systems have typically used different acoustic encoders. Recent work has attempted to unify them by either jointly training a fixed stack of streaming and non-streaming layers or using knowledge distillation during training to ensure consistency between the streaming and non-streaming predictions. We propose mixture model (MiMo) attention as a simpler and theoretically-motivated alternative that replaces only the attention mechanism, requires no change to the training loss, and allows greater flexibility of switching between streaming and non-streaming mode during inference. Our experiments on the public Librispeech data set and a few Indic language data sets show that MiMo attention endows a single ASR model with the ability to operate in both streaming and non-streaming modes without any overhead and without significant loss in accuracy compared to separately-trained streaming and non-streaming models. We also illustrate this benefit of MiMo attention in a second-pass rescoring setting.

Index Terms: attention, mixture models, recurrent neural network transducer, end-to-end speech recognition.

1. Introduction

End-to-end and all-neural automatic speech recognition (ASR) has led to rapid strides in word error rate (WER) across a variety of tasks and languages over the past several years. Such systems combine the acoustic model/encoder and language model into a single system, in contrast to earlier *hybrid* ASR systems that separated these models. Promiment examples include models trained with connectionist temporal classification (CTC) loss [1], recurrent neural network transducer (RNN-T) loss [2], and listen, attend, and spell (LAS) models [3]. Attention-based acoustic encoders such as Transformers [4] and the more recent Conformers [5] have become more popular over recurrent neural networks and long short-term memory (LSTM) [6] networks. This paper focuses on such attention-based acoustic encoders trained within the RNN-T model.

While the language model/decoder in these end-to-end systems operates from left to right, the acoustic encoder can be either *streaming* (strict left-to-right) [7, 8, 9, 10] or *non-streaming* [11, 12, 13]. As the name suggests, streaming encoders are used for real-time transcription with strict latency constraints, and can only use acoustic information from time steps $l \leq t$ to produce the acoustic embedding at time step t. In case of Conformers or Transformers, this means that the attention probability density function (PDF) at time t of the output sequence is constrained to be zero $\forall l > t$. On the other hand, non-streaming acoustic encoders can potentially utilize the entire speech utterance to produce the acoustic embeddings at each time step. Thus, a non-streaming Conformer or Transformer can have a non-zero attention PDF over the entire input sequence. Such non-streaming acoustic encoders are suitable for offline transcription, and typically offer lower WERs than their comparable streaming counterparts.

Conventionally, the choice of streaming and non-streaming acoustic encoders was dictated by the specific use-case and these models were trained separately. Recent work has attempted to unify these two types of acoustic encoders to reduce the ASR model inventory. The *cascade* encoder [14] trains a stack of N_{stream} streaming encoder layers at the bottom and $N_{\text{nonstream}}$ non-streaming encoder layers at the top. The choice of values of N_{stream} and $N_{\text{nonstream}}$ is made at training time. The outputs of both the final streaming and final non-streaming layers are used to compute separate streaming and non-streaming RNN-T losses. Training proceeds by minimizing a weighted sum of these two losses. The *dual-mode* encoder [15] shares all the encoder layers and computes predictions for both streaming (left context attention) and non-streaming (left and right context attention) modes during training. The training loss is a sum of the streaming loss, non-streaming loss, and a knowledge distillation loss penalizing large deviations between the streaming and non-streaming predictions. The authors showed that knowledge distillation was especially crucial to obtaining good WER. The *Y-model* [16] also adopts a similar stack of streaming and non-streaming layers, where the non-streaming layers are trained with randomly-sampled right context. Attempts to unify streaming and non-streaming architectures with two passes using a hybrid CTC/attention architecture [17] and shared encoders for the two passes with dynamic latency control [18] have also been proposed recently.

We hypothesize that the fundamental limitation of the acoustic encoder lies in the computation of the attention PDF. Specifically, using a single softmax over the entire context window constrains the model to use the same context window size both during training and inference. We quantify the error resulting from using a smaller context window for softmax computation during inference. We propose Mixture Model (MiMo) attention as a flexible alternative. MiMo attention computes the attention PDF using a mixture model of softmaxes over the context window. The support of the mixture components can be designed to cover all possible use cases during inference, e.g. whole-sequence context, limited left+right context, and left-only context. MiMo attention does not add any complexity or parameters to the model. Since each mixture component is a fully-normalized softmax, we can use any subset of mixture components during inference without any mismatch. Hence, sequence-to-sequence models trained with MiMo attention allow both streaming and non-streaming modes during inference.

We begin with the introduction of MiMo attention in Section 2. Section 3 presents ASR experiments on LibriSpeech and

Indic language data sets. We also illustrate the benefit of MiMo attention in a second-pass rescoring setting with the Neural Oracle Search (NOS) [19], where the same model is used with left context in the first pass and provides left+right context encoder activations for second pass rescoring. Visualization of MiMo attention (Section 4) and conclusion (Section 5) follow.

2. MiMo Attention

2.1. Conventional Attention

We first discuss the conventional attention mechanism pervasive in literature. Consider an length-T sequence $(x_0, \ldots, x_{T-1})$ of input feature vectors. We would like to output a sequence $(y_0, \ldots, y_{T-1})$ of feature vectors using an attention mechanism. Consider the generation of the k^{th} vector y_k in the sequence. Let $(s_0^k, \ldots, s_{T-1}^k)$ denote the unnormalized attention scores for the T time steps that we will use for generating y_k. Our discussion does not depend on how these scores were generated. For example, these scores could have been computed by taking the dot product between a *query* vector for time step k with all the T *key* vectors corresponding to the input sequence.

We first convert these attention scores to a PDF over the T time steps using the softmax function:

$$(p_0^k, \ldots, p_{T-1}^k) = \text{softmax}(s_0^k, \ldots, s_{T-1}^k) \ . \qquad (1)$$

Next, we compute the expectation of T *value* vectors $(v_0, \ldots, v_{T-1})$ using the above attention PDF to produce y_k:

$$y_k = \sum_{t=0}^{T-1} p_t^k v_t \ . \qquad (2)$$

The intuition is that the attention probability p_t^k captures the importance of the value vector v_t while computing y_k. The output feature vector y_k provides a fixed-length summary of the entire length-T input sequence. In the standard self-attention [20] used in Transformers and Conformers, the key, query and value vectors are all learned linear transforms of the input vector x_t.

2.2. Attention Probability Error in Conventional Attention

In order to understand the limitation of the conventional single-softmax attention, we next consider the situation when the context window is reduced to $T_{\text{infer}} < T$ at inference. For instance, training might use a context size T spanning both left and right context, but inference may only allow a left context of size T_{infer} and zero right context due to latency considerations. This results in the following truncated softmax computation during inference:

$$\hat{p}_t^k = \frac{\exp(s_t^k)}{\sum_{l=0}^{T_{\text{infer}}-1} \exp(s_l^k)} \text{ for } t \leq T_{\text{infer}} \text{ and 0 otherwise} \ . \qquad (3)$$

It is easy to see that $\hat{p}_t^k$ overestimates the attention probability with respect to the correct value p_t^k for $t < T_{\text{infer}}$. This is because

$$\hat{p}_t^k - p_t^k = \frac{Z(T_{\text{infer}} : T)}{Z(0 : T_{\text{infer}}) Z(0 : T)} \exp(s_t^k) \geq 0 \qquad (4)$$

where $Z(T_{\text{infer}} : T)$ denotes the softmax denominator sum from T_{infer} to $T - 1$. Similarly, it is easy to see that $\hat{p}_t^k$ underestimates the attention probability for $t \geq T_{\text{infer}}$.

The above estimation error is a key limitation of standard single-softmax attention that leads to performance regression when $T_{\text{infer}} \neq T$. The next section introduces MiMo attention as a means to eliminate this error and describes a special case of interest in this paper.

2.3. MiMo Attention

In place of the single softmax in (1), MiMo uses a mixture of $M \geq 1$ softmaxes to compute the attention PDF:

$$(p_0^k, \ldots, p_{T-1}^k) = \sum_{m=0}^{M-1} w_m \text{softmax}_m(s_0^k, \ldots, s_{T-1}^k) \ , \qquad (5)$$

where $(w_0, \ldots, w_{M-1})$ are the mixture weights that are non-negative and sum to 1. The key advantage of using a mixture model is that the support of the mixture components can be picked to cater to all possible contexts during inference. During inference, one can simply set the mixture weight corresponding to the desired context to 1 and the rest to 0, with the guarantee of a properly normalized attention PDF. This flexibility is not available in standard single-softmax attention.

Consider the special case of a left context of L time steps and a right context of R time steps relative to the center frame. We construct MiMo attention with $M = 2$ mixture components, where the first component spans over $[k-L, k]$ time steps and the second component spans over $[k+1, k+R]$, where k is the center frame index. In other words, the first softmax operates over the left+center context whereas the second softmax operates over the right context. Figure 1 illustrates an attention probability matrix of size $T \times T$ with probabilities outside the valid context set to 0. MiMo attention splits this band diagonal matrix into a weighted sum of a lower-band diagonal matrix for the left+center context and an upper-band diagonal matrix for the right context. The two matrices are row-stochastic and hence represent properly-normalized attention PDFs.

During inference, one can use left+right context (non-streaming mode) by using non-zero values of both mixture weights, or left-only context (streaming mode) by setting the mixture weights $w_0 = 1$ and $w_1 = 0$. It is easy to see that MiMo attention is a general framework that can accommodate all types of contexts and use-cases during inference. In contrast to prior work, training an ASR model with MiMo attention does not change the training loss or add any additional complexity.

2.4. MiMo Weight Noise

We first fixed $w_0 = w_1 = 0.5$ during training, but found it to be useful to add noise to the mixture weights. We hypothesize that this mixture weight noise improves the generalization of MiMo. We experimented with two types of additive noise – Uniform and Bernoulli for the $M = 2$ case. In uniform noise, we sample $u \sim \text{uniform}(0, w_1)$ per training batch and use it to perturb the mixture weights as $w_0 + u$ and $w_1 - u$. In Bernoulli noise, we sample $u \sim \text{Bernoulli}(0, w_1)$. We compare these various options on Librispeech in the next section.

3. Experiments and Results

3.1. Data

We use two very different corpora for evaluating the proposed algorithm, namely the well-benchmarked, publicly-available Librispeech [21] corpus and an in-house data set representative of Google's voice search traffic. The Librispeech training corpus comprises of 960 hours of speech from over 2000 speakers. The in-house ASR training data from short voice

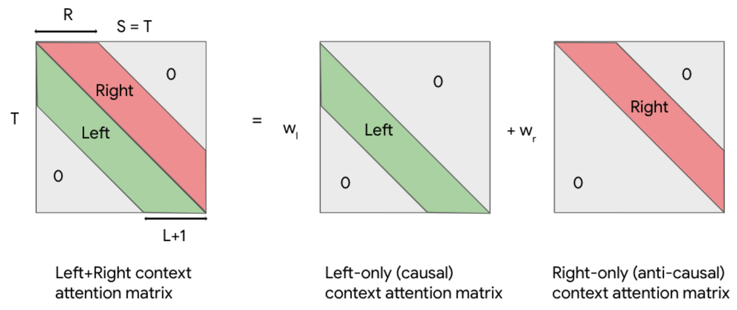
R
S = T
T
Right
Left
0
0
L+1
=
w_l
Left
0
0
+ w_r
Right
0
0
Left+Right context attention matrix
Left-only (causal) context attention matrix
Right-only (anti-causal) context attention matrix

Figure 1: *This figure illustrates MiMo attention for the simple case of $M = 2$ mixture components. The $T \times T$ attention probability matrix on the left is defined as a mixture model composed of left-only (causal) and right-only (anti-causal) attention matrices. All matrices are row-stochastic, i.e. each row sums to 1.*

search utterances for all languages are anonymized and hand-transcribed, and are representative of Google's voice search traffic. The training set is created by artificially corrupting clean utterances using a room simulator and the addition of various noise styles [22]. The training data for the Indic languages presented in this paper, namely Gujarati and Tamil, contains 4.9M and 5.2M million utterances respectively. The development set is a small fraction of the training set held out for validation. The test set also contains anonymous transcribed utterances from the voice-search task. The WER metric we report is transliteration-optimized WER described in [23] to accommodate mixed writing scripts frequently seen in Indic languages.

3.2. Model

The conformer model follows an encoder-decoder architecture detailed in [5] and comprises of a 17-layer conformer acoustic encoder with self-attention in all layers, and a 2-layer LSTM decoder trained using the RNN-T loss. We use a model dimension of 512 for the encoder, which gives a total of 118M parameters. The input acoustic features for our LibriSpeech experiments are the same as used in [5]. For our Indic experiments, the input acoustic features to the model are 80-dimensional log-Mel features stacked over three frames as described in [24]. The target vocabulary for all models are wordpieces with a vocabulary of 1024 for Librispeech and 4096 for Indics. All models are randomly-initialized and trained with an effective batch size of 4096 in Lingvo [25] on Tensor Processing Unit slices.

3.3. Librispeech Experiments

We first describe our experiments on the public 960-hour Librispeech data set. All checkpoints were picked based on the dev-clean WER. We trained two baseline models:

1. **Base-(64, 0)**: a left context-only model with context size of 64 frames.
2. **Base-(64, 64)**: a non-streaming model with left context of 64 and right context of 64 frames.

Unlike the state-of-the-art Librispeech system [26] we did not use any LM or additional training corpora. Table 1 shows the WERs on test-clean and test-other for the Librispeech-960 data set. The Base-(64, 64) model gives a lower WER than the streaming Base-(64, 0) model when using the matched non-streaming (64, 64) inference context. However, its WER nearly doubles on both test-clean and test-other when using a streaming (64, 0) inference context.

Table 1: *WERs of baseline and MiMo Conformers on Librispeech-960. (L, R) denotes the size of (left, right) context.*

Model	Inference Context	test-clean WER	test-other WER
Base-(64, 0)	(64, 0)	2.5	5.9
Base-(64, 64)	(64, 64)	2.2	5.2
	(64, 0)	4.4	11.1
MiMo-(64, 64)	(64, 64)	2.3	5.2
	(64, 0)	3.6	9.1
+ uniform noise	(64, 64)	2.4	5.7
	(64, 0)	2.7	6.7
+ Bernoulli noise	(64, 64)	2.5	5.8
	(64, 0)	3.0	7.6

We first experimented with MiMo without mixture weight noise. As shown in Table 1, it matches the WER of the Base-(64, 64) model when using the matched inference context. When we switched the inference context to (64, 0), the WER rises as expected, but the increase is significantly lower compared to the Base-(64, 64) model. In fact, MiMo-(64, 64) gives nearly 18% relative lower WER on both test-clean and test-other compared to the Base-(64, 64) model when operating with a (64, 0) streaming inference context.

Next, we introduced mixture weight noise during training of the MiMo Conformer model. Table 1 shows the impact of both uniform and Bernoulli noise. We observed that this increases the WERs from (2.3, 5.2) to (2.4, 5.7) on (test-clean, test-other) for uniform noise and to (2.5, 5.8) for Bernoulli noise when using a (64, 64) inference context. However, we obtain significant WER improvement from (3.6, 9.1) to (2.7, 6.7) for uniform noise when running MiMo with the streaming (64, 0) inference context. Similar gains were observed for Bernoulli noise. We conclude that despite MiMo's ability to provide a correctly normalized attention PDF for a (64, 0) streaming graph, weight noise regularizes the model by exposing it to a variety of mixture weights during training. Uniform noise performs better than Bernoulli noise in all conditions. All further MiMo experiments in this paper include uniform weight noise.

3.4. Indics First and Second Pass Experiments

We present results on two Indic languages – Gujarati and Tamil in Table 2. We observe a similar trend as our Librispeech exper-

Table 2: *WERs of baseline and MiMo streaming/non-streaming Conformers on Indic voice search data.*

Model	Inference Context	Gujarati WER	Tamil WER
Base-(64, 0)	(64, 0)	22.3	22.3
Base-(64, 64)	(64, 64)	21.3	20.0
	(64, 0)	36.5	33.2
MiMo-(64, 64)	(64, 64)	21.9	20.7
	(64, 0)	21.5	22.0

Table 3: *First and second pass WERs of baseline and MiMo Conformers on Tamil voice search data.*

Model	First-Pass		Second-pass	
	Inference Context	WER	Inference Context	WER
Base-(64, 0)	(64, 0)	22.3	(64, 0)	21.1
Base-(64, 64)	(64, 64)	20.0	(64, 64)	19.6
MiMo-(64, 64)	(64, 64)	20.7	(64, 64)	20.2
	(64, 0)	22.0	(64, 0)	21.4
			(64, 64)	20.9

iments. While the Base-(64, 64) model suffers significant WER regression (71.4% for Gujarati and 66% for Tamil) upon using a streaming (64, 0) inference context, the MiMo-(64, 64) model suffers none to a small regression in WER.

Next, we experimented with second-pass rescoring via the Neural Oracle Search (NOS) algorithm [19]. NOS trains a small neural network that takes the N-best hypotheses, acoustic encoder activations, and external LM score as input and re-ranks the N-best hypotheses. The combination of NOS with MiMo is especially interesting since one can run MiMo with streaming (64, 0) context during the first pass to generate N-best hypotheses, and then use the (64, 64) context to derive encoder activations for second pass rescoring. This is not possible in the non-MiMo baseline models since we are constrained to use matched first and second pass inference contexts.

Table 3 shows the WERs for various Tamil ASR models after first-pass and second-pass rescoring. We show matched inference contexts during first and second pass for the two baseline models. As expected, we see WER improvement upon using second-pass rescoring. For the single MiMo model, we report the two matched first/second pass WERs, and also the situation where the first pass uses context (64, 0) and the second pass uses acoustic encoder activations from the (64, 64) context. It is interesting to observe that changing the encoder activations from (64, 0) to (64, 64) despite still using N-best hypothesis from (64, 0) in the first pass gives an improvement in WER from 21.4% to 20.9%.

4. Visualization of MiMo attention

Given the well-documented sharpness of posterior PDFs from end-to-end ASR models, it is natural to ask whether a mixture model is an overkill for modeling self-attention PDFs. Figure 2 shows the alpha (forward), beta (backward), and gamma (state-occupancy/“attention”) PDFs computed during the RNNT loss calculation for a Librispeech training utterance. As expected, the gamma PDF is quite sharp, i.e. has low entropy. In contrast, Figure 3 shows the self-attention probability matrix of one attention head from the 1st and 17th layers of the conformer acoustic encoder trained with MiMo. Each row of these attention matrices are modeled by a mixture of two softmaxes corresponding to the left and right of the main diagonal. We note that there are several frames in each row that contain significant probability mass, not just one. These frames appear as dark vertical lines in the attention plot, indicating high attention PDF value. Further, the probability mass is not simply concentrated on the main diagonal, indicating that frames several steps apart in time also contribute to self-attention. These observations lend support to use of a mixture model for the self-attention PDFs.

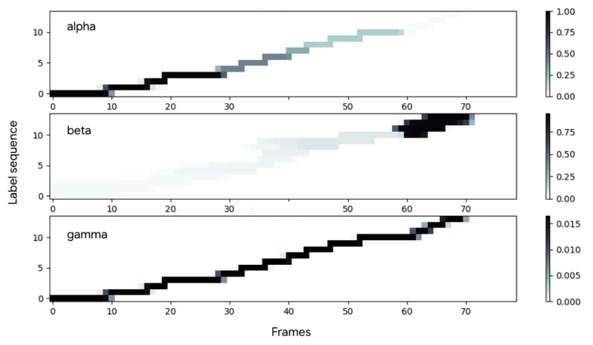

Figure 2: *alpha, beta, and gamma used in RNNT loss computation for a Librispeech training utterance.*

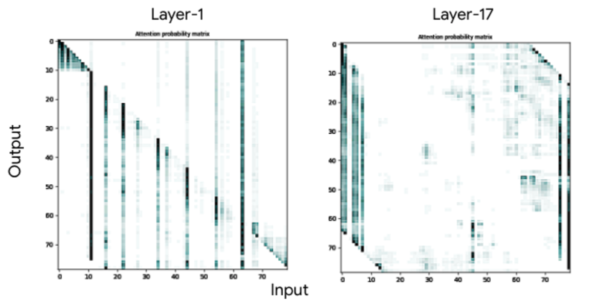

Figure 3: *MiMo attention probability matrices from the first attention head of the 1st and 17th conformer layers using a Librispeech training utterance. The choice of layers was arbitrary.*

5. Conclusion

This paper presents mixture model (MiMo) attention for conformer and transformer-based end-to-end automatic speech recognition. MiMo attention enables training a single ASR model that can be used in both streaming (left context-only) and non-streaming (left and right context) modes during inference. This is possible because MiMo attention uses a mixture of softmaxes to model the attention PDF instead of a single softmax. This mixture model can be designed to cater to all possible conditions during inference, to ensure a properly-normalized attention PDF for both streaming and non-streaming inference. Our experiments on Librispeech and two in-house Indics data sets show the benefit and flexibility of MiMo attention. The power of MiMo is further illustrated through second-pass rescoring experiments, wherein we can use encoder activation from left+right context to rescore first-pass hypotheses derived using left-only context.

6. References

[1] 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.

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

[3] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *arXiv preprint arXiv:1508.01211*, 2015.

[4] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7829–7833.

[5] 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," *arXiv preprint arXiv:2005.08100*, 2020.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] Y. He, T. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S. yiin Chang, K. Rao, and A. Gruenstein, "Streaming end-to-end speech recognition for mobile devices," 2019. [Online]. Available: https://arxiv.org/abs/1811.06621

[8] H. Inaguma, Y. Gaur, L. Lu, J. Li, and Y. Gong, "Minimum latency training strategies for streaming sequence-to-sequence ASR," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6064–6068.

[9] X. Chen, Y. Wu, Z. Wang, S. Liu, and J. Li, "Developing real-time streaming transformer transducer for speech recognition on large-scale dataset," *arXiv preprint arXiv:2010.11395*, 2020.

[10] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and LSTMs," *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 373–377, 2017.

[11] Z. Tüske, K. Audhkhasi, and G. Saon, "Advancing sequence-to-sequence based speech recognition." in *INTERSPEECH*, 2019, pp. 3780–3784.

[12] Z. Tüske, G. Saon, K. Audhkhasi, and B. Kingsbury, "Single headed attention based sequence-to-sequence model for state-of-the-art results on Switchboard-300," *arXiv preprint arXiv:2001.07263*, 2020.

[13] K. Audhkhasi, G. Saon, Z. Tüske, B. Kingsbury, and M. Picheny, "Forget a Bit to Learn Better: Soft Forgetting for CTC-Based Automatic Speech Recognition." in *INTERSPEECH*, 2019, pp. 2618–2622.

[14] A. Narayanan, T. N. Sainath, R. Pang, J. Yu, C.-C. Chiu, R. Prabhavalkar, E. Variani, and T. Strohman, "Cascaded encoders for unifying streaming and non-streaming ASR," *arXiv preprint arXiv:2010.14606*, 2020.

[15] J. Yu, W. Han, A. Gulati, C.-C. Chiu, B. Li, T. N. Sainath, Y. Wu, and R. Pang, "Dual-mode ASR: Unify and improve streaming ASR with full-context modeling," *Proceedings of ICLR*, 2021.

[16] A. Tripathi, J. Kim, Q. Zhang, H. Lu, and H. Sak, "Transformer transducer: One model unifying streaming and non-streaming speech recognition," *arXiv preprint arXiv:2010.03192*, 2020.

[17] B. Zhang, D. Wu, Z. Yao, X. Wang, F. Yu, C. Yang, L. Guo, Y. Hu, L. Xie, and X. Lei, "Unified streaming and non-streaming two-pass end-to-end model for speech recognition," *arXiv preprint arXiv:2012.05481*, 2020.

[18] Z. Gao, S. Zhang, M. Lei, and I. McLoughlin, "Universal ASR: Unifying Streaming and Non-Streaming ASR Using a Single Encoder-Decoder Model," *arXiv preprint arXiv:2010.14099*, 2020.

[19] E. Variani, T. Chen, J. Apfel, B. Ramabhadran, S. Lee, and P. Moreno, "Neural Oracle Search on N-BEST Hypotheses," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7824–7828.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[21] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[22] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. Sainath, and M. Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *INTERSPEECH*, 2017.

[23] J. Emond, B. Ramabhadran, B. Roark, P. Moreno, and M. Ma, "Transliteration based approaches to improve code-switched speech recognition performance," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 448–455.

[24] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv preprint arXiv:1507.06947*, 2015.

[25] J. Shen, P. Nguyen, Y. Wu, Z. Chen, M. X. Chen, Y. Jia, A. Kannan, T. Sainath, Y. Cao, C.-C. Chiu *et al.*, "Lingvo: a modular and scalable framework for sequence-to-sequence modeling," *arXiv preprint arXiv:1902.08295*, 2019.

[26] D. S. Park, Y. Zhang, Y. Jia, W. Han, C.-C. Chiu, B. Li, Y. Wu, and Q. V. Le, "Improved noisy student training for automatic speech recognition," *arXiv preprint arXiv:2005.09629*, 2020.