Weak-Attention Suppression For Transformer Based Speech Recognition

Yangyang Shi, Yongqiang Wang, Chunyang Wu, Christian Fuegen, Frank Zhang, Duc Le, Ching-Feng Yeh, Michael L. Seltzer

Facebook AI, USA

`{yyshi,yqw,chunyang,fuegen,frankz,duchoangle,cfyeh,mikeseltzer}@fb.com`

Abstract

Transformers, originally proposed for natural language processing (NLP) tasks, have recently achieved great success in automatic speech recognition (ASR). However, adjacent acoustic units (i.e., frames) are highly correlated, and long-distance dependencies between them are weak, unlike text units. It suggests that ASR will likely benefit from sparse and localized attention. In this paper, we propose Weak-Attention Suppression (WAS), a method that dynamically induces sparsity in attention probabilities. We demonstrate that WAS leads to consistent Word Error Rate (WER) improvement over strong transformer baselines. On the widely used LibriSpeech benchmark, our proposed method reduced WER by 10% on `test-clean` and 5% on `test-other` for streamable transformers, resulting in a new state-of-the-art among streaming models. Further analysis shows that WAS learns to suppress attention of non-critical and redundant continuous acoustic frames, and is more likely to suppress past frames rather than future ones. It indicates the importance of lookahead in attention-based ASR models.

Index Terms: automatic speech recognition, transformer, weak-attention suppression

1. Introduction

In recent years, models based on transformers [1] and their variants have achieved state-of-the-art results in many natural language processing (NLP) tasks, such as language modeling, machine translation, and natural language understanding [2, 3, 4]. The transformer relies on the multi-head self-attention method, eschewing the recurrent connection in recurrent neural networks. The attention method connects arbitrary positions in the whole sequence directly, allowing the model to capture long-range dependencies regardless of distance. Each transformer layer performs computations for the whole sequence in parallel, thus improving efficiency.

In automatic speech recognition (ASR), transformer-based architectures have also showed superior performance across various modeling paradigms, including sequence-to-sequence [5, 6, 7, 8], neural transducer [9, 10], as well as traditional hybrid [11, 12] and CTC [13] systems.

While similar in some aspects, ASR tasks are very different from many NLP tasks. ASR systems extract the input features from continuous signals rather than discrete text units. Many acoustic units (i.e., frames) are typically needed to convey a single text unit's semantic meaning. Many of these acoustic units are not critical in long-range dependencies, for example, silence. Based on these observations, we propose the Weak-Attention Suppression (WAS) method to improve transformer-based models for ASR. The method aims to induce sparsity in the attention probability distribution by dynamically determining a threshold for each time frame. All attention probabilities smaller than this threshold are set to zero, and the remaining probabilities are re-normalized to sum to one.

To verify the performance of the proposed method, we carry out experiments on the widely used LibriSpeech corpus [14]. WAS consistently improves Word Error Rate (WER) of transformer-based hybrid acoustic models across different model sizes (12-layer 40M parameters vs. 24-layer 81M parameters) and variants (non-streamable full context [12] vs. streamable limited context [15]). The relative WER improvement ranges from 6% to 10% on `test-clean` and 2% to 6% on `test-other`, with larger impact on streamable transformers. Our results with WAS and streamable transformers established a new state-of-the-art on Librispeech among streaming models to the best of our knowledge.

A more in-depth analysis of individual utterances and the whole dev-clean split in LibriSpeech data shows that WAS suppresses attention from non-critical acoustic units, such as silence. For adjacent acoustic units, attention from redundant units is suppressed, especially for the lower layers. The analysis also demonstrates that past acoustic units are more likely to be suppressed than future ones. In other words, lookahead is especially crucial for transformer-based acoustic models.

2. Related Work

The proposed WAS method aims to improve transformer-based acoustic models in hybrid speech recognition for both streaming [11, 15], and non-streaming [12] applications.

For the non-streaming case, we use the same model architecture proposed in [12], where the whole model architecture consists of three parts, convolution layers, transformer layers, and auxiliary intermediate layer losses. The convolution layers play a similar role as another positional encoding method [16]. A stack of transformers with both multi-head self-attention and feed-forward network (FFN) is used on top of the convolution layers. To effectively train the deep transformers, the final loss function is interpolated with auxiliary intermediate layer losses [17] that pass the gradients to the intermediate layers in the deep structure.

To extend transformer-based acoustic models for streaming applications, the work [15] applies block processing to segment the whole utterance into multiple chunks with lookahead context. To carry over information across chunks, they modify the self-attention with the augmented memory bank. Each slot in the augmented memory bank stores the embedding vectors for previous chunks.

Sparse attention in the transformer has been explored in [18] to reduce model inference complexity. Unlike this work, our method aims to improve ASR accuracy by inducing sparsity in both training and inference.

Different from the works in [19, 20] that achieve sparse attention by replacing the softmax function with α-entmax or sparsemax, our method keeps the softmax function and suppresses the attention based on the probability distribution. Comparing these different methods for suppressing attention will be an interesting topic for future research.

3. Transformer Based Acoustic Model With Weak-Attention Suppression

Before explaining the details of the WAS, the following subsection gives a short description of multi-head self-attention.

3.1. Multi-Head Self Attention

Given the input embedding sequence $\boldsymbol{X} \in \mathbb{R}^{L \times d_i}$ with sequence length L, the projection matrices $\mathbf{W}_{\mathrm{q}}$, $\mathbf{W}_{\mathrm{k}}$ and $\mathbf{W}_{\mathrm{v}}$ transform the $\boldsymbol{X}$ into *query*, *key* and *value* space, respectively.

$$\boldsymbol{Q} = \boldsymbol{X}\mathbf{W}_{\mathrm{q}}, \boldsymbol{K} = \boldsymbol{X}\mathbf{W}_{\mathrm{k}}, \boldsymbol{V} = \boldsymbol{X}\mathbf{W}_{\mathrm{v}}. \tag{1}$$

The attention probabilities are computed in dot-product way as follows:

$$\boldsymbol{A} = \mathrm{softmax}(\frac{\boldsymbol{K}\boldsymbol{Q}^T}{\sqrt{d_i}}), \tag{2}$$

where $\boldsymbol{A}$ is a $L \times L$ matrix. Each element $\alpha_{i,j}$ represents the attention probability of the *query* at i position with the *key* at j position.

Given $\boldsymbol{A}$, the output embedding sequence of self-attention is obtained via:

$$\boldsymbol{Z} = \mathrm{Dropout}(\boldsymbol{A})\boldsymbol{V}. \tag{3}$$

Rather than performing a single self-attention function with queries, keys and values, it was found beneficial to run self-attention h times in parallel with different projection matrices $\mathbf{W}_{\mathrm{q}}$, $\mathbf{W}_{\mathrm{k}}$ and $\mathbf{W}_{\mathrm{v}}$. The outputs from each self-attention are concatenated and linearly projected, resulting to final output, i.e.,

$$\boldsymbol{O} = \mathbf{W}_{\mathrm{o}}\mathrm{Concate}(\boldsymbol{Z}^1, ..., \boldsymbol{Z}^h) \tag{4}$$

where $\mathbf{W}_{\mathrm{o}} \in \mathbb{R}^{d_i \times h d_{\mathrm{v}}}$, and $\boldsymbol{Z}^i$ is the output from the i-th self-attention.

3.2. Weak-Attention Suppression

In speech recognition tasks, many acoustic units, e.g., silence, may not play a critical role in long-distance dependencies. For continuous acoustic units that share similarities, some of them may be redundant for long-distance dependencies. Comparing with many NLP tasks, the number of acoustic units in audio utterances are much more significant than the number of text units in sentences. Due to these factors, the sparse attention is more desirable for speech recognition. However, in self-attention, the softmax function is used to get the attention probabilities. One limitation of softmax function is that it always generates dense attention, i.e., $\mathrm{softmax}(X)_i \neq 0$; all the elements of the softmax function is over zero.

The WAS sets the attention probabilities smaller than a threshold to zero and normalizes the rest attention probabilities. The threshold is determined based on the following scheme:

$$\theta_i = m_i - \gamma\delta_i \tag{5}$$

where m_i and δ_i are the mean and standard deviation of the attention probability for i-th position in *query*, respectively; γ is a user-specified hyper-parameter. So the threshold θ_i can be represented as

$$\theta_i = \frac{1}{L} - \gamma\sqrt{\frac{\sum_{j=1}^{L}(\alpha_{i,j} - \frac{1}{L})^2}{L-1}}, \tag{6}$$

where L is the length of *key* in self-attention. The attention probability $\alpha_{i,j}$ less than θ_i, is set to zero. The rest non-zero attention probabilities are re-normalized.

In practical implementation, we realize the re-normalization in two steps. Firstly, using softmax function on attention weights, we get the attention probabilities. Based on formula (6), we replace the attention weight to negative infinity when its attention probability is lower than the threshold. Applying the softmax function on the processed attention weight generates the re-normalized attention probabilities.

4. Experiments

4.1. Data

To verify the performance of the proposed method, we conduct experiments on the LibriSpeech corpus [14]. LibriSpeech is an open-source speech corpus that contains 1000 hours of speech derived from audiobooks in the LibriVox project. The audio data [1], language model data, and pre-trained language models [2] are available for downloading. The development data and evaluation data in LibriSpeech are split into simple "clean" subsets and more difficult "other" subsets. We use the official 4-gram language model for decoding in all experiments.

4.2. Setup

In all experiments, we use context and positional dependent graphemes (i.e., chenones) as output units [21]. We bootstrap the HMM-GMM system following the standard Kaldi [22] LibriSpeech recipe. For feature extraction, we use 80-dimensional logMel filter bank features at a 10ms frame rate. To increase the training robustness, both speed perturbation [23] and *SpecAugment* [24] without time warping are used.

We apply the same model architecture as [12, 15], which has two VGG blocks [25] followed by a stack of transformer layers. Each VGG block has two consecutive 3-by-3 convolution layers followed by a Relu activation function and a max-pooling layer. While the first VGG block uses 32 channels, the second VGG block uses 64 channels. Both VGG blocks use a 2-by-2 Max-pooling. The first VGG block uses stride 2; the second one uses stride 1. Overall, two VGG blocks have stride 2. From an input sequence of 80-dim feature vector at a 10ms rate, the VGG blocks generate a 2560-dim feature vector sequence at a 20ms rate.

Each transformer layer applies 8 heads of self-attention. The dimensionality of input and output for each transformer layer is 512, and the inner-layer of FFN has dimensionality 2048. We experiment with different model sizes (12 transformer layers, and 24 transformer layers architectures) to verify the improvement from the WAS. To overcome the training divergence issue for deep transformer models, e.g., 24 transformer layers, we apply an auxiliary incremental loss [17]. A linear transformation, together with a Relu function, projects the outputs from the 6/12/18-th transformer layers to the final output space. The auxiliary incremental loss takes the projected

[1]http://www.openslr.org/12/
[2]http://www.openslr.org/11/

outputs to compute the auxiliary CE losses interpolated with the original CE loss with a 0.3 weight.

In all experiments, we use the adam optimizer [26] with a tri-stages learning-rate strategy, i.e., warming-up stage, holding stage, and decaying stage. 8K updates in the warming-up stage increase the learning rate from 1e-5 to 1e-3 for non-streaming transformer-based models and 3e-4 for streaming transformer-based models, respectively. The holding stage uses 100K updates. The decaying stage exponentially decreases the learning rate to 1e-5.

To efficiently use GPU resources, the batch size is dynamically determined. Each batch contains around 10,000 to 20,000 frames, including padding frames. We train all the models using 32 Nvidia V100 GPUs. To train the transformer-based model effectively, we segment the training utterances into less than 10 seconds using forced alignment results from an existing latency-controlled BLSTM acoustic model. We select the best model based on WER on the dev set and report its result on test data.

4.3. Results

Model Arch	#Params (M)	test-clean	test-other
BLSTM[12]	79	3.11	7.44
vggBLSTM[12]	95	2.99	6.95
vggTrf-12	40	3.11	7.14
vggTrf-12-WAS	40	2.93	6.73
vggTrf-24[12]	81	2.66	5.64
vggTrf-24-WAS	81	2.50	5.55
AmTrf-24[15]	81	3.09	7.08
AmTrf-24-WAS	81	2.78	6.71

Table 1: *WER comparison on the LibriSpeech benchmark. For models with citations, we directly use the results from the referred paper. '-WAS' means the weak-attention suppression with $\gamma = 0.5$ is applied. Note results in this table are obtained without the NNLM rescoring.*

Table 1 gives the WER comparison of different hybrid models on LibriSpeech data. Overall, the transformer-based models show improvement over recurrent neural network counterparts. For small transformer based model 'vggTrf-12', applying WAS brings a relative WER reduction of 5.8% on test-clean and 5.7% on test-other. On a large model with 24 transformer layers, WAS generates a relative WER reduction of 6.0% on test-clean and 5.7% on test-other. 'AmTrf-24' is a streaming transformer-based model using augmented memory that limited the attention within an audio segment with 320ms lookahead. The WAS achieves relative WER reduction 10.0% on test-clean and 5.2% on test-other over the streaming baseline.

Model	γ	test-clean	test-other
vggTrf-12	-	3.11	7.14
vggTrf-12-WAS	0.0	2.97	7.13
vggTrf-12-WAS	0.5	2.93	6.73
vggTrf-12-WAS	1.0	3.04	6.92

Table 2: *Word error rate comparison of different γ selection in weak-attention suppression on LibriSpeech data.*

Table 2 gives the WER results from different γ selection on LibriSpeech data using the small size of the transformer model. $\gamma = 0.5$ gives us the best results. We constrain the γ selection from 0 to 1. If the attention probability is close to a normal distribution, we suppress roughly 16% to 50% the attention. According to our analysis, when $\gamma = 0.5$, 36% of attention got suppressed in the first transformer layer.

4.4. Attention Suppression Analysis

In this section, we analyze the effect of WAS in one individual utterance and the whole dev-clean split in LibriSpeech using the model 'vggTrf-12-WAS' in Table 1.

In individual utterance analysis, we average the portion of the suppressed attention over the multiple heads and the whole input sequence for each transformer layer. The following formula represents the processing.

$$s_{i,j}^k = \begin{cases} 1 & \alpha_{i,j}^k < \theta_i^k \\ 0 & \alpha_{i,j}^k \geq \theta_i^k \end{cases} \tag{7}$$

where $\alpha_{i,j}^k$ represents the attention probability of the i-th position in *query* with j-th position in *key* from the k-th head. θ_i^k is the suppression threshold for i-th position in *query* in the k-th head. The threshold is determined according to formula (6).

$$f(j) = \frac{\sum_{i=1}^{i=L} \sum_{k=1}^{k=H} s_{i,i}^k}{LH} \tag{8}$$

where L is the length of *key* and H the number heads in multi-head self-attention. The function value $f(j)$ indicates the weakness of the attention at the j-th position in the *key*.

Figure 1(b),1(c) and 1(d) draws the function $y = s(j)$ for 1st, 6-th and 12-th transformer layer. The horizontal axis stands for the position in the *key*. The vertical axis stands for the averaged portion of suppressed attention. Note 'vggTrf-12-WAS' uses VGG blocks with stride 2. For one-second audio using frameshift 10ms, the length of the input sequence to each transformer layer is 500.

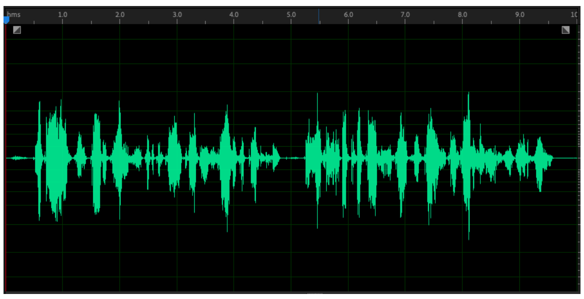
By comparing the audio wave 1(a) with 1(b), it is obvious to see there are three peaks (at the beginning, in the middle and at the end) in 1(b) that are the exact position of a long silence in the audio wave. The rest of the small peaks in 1(b) seem to be in the rough location of short silence in the audio wave in 1(a). This phenomenon indicates that using weak-attention suppression in the first transformer layer can suppress the attention to silence.

Zooming into Figure 1(b) and Figure 1(c), there are dramatic ups-and-downs even for the non-silence consecutive positions in the input sequence that are supposed to share acoustic similarities. This phenomenon suggests that for consecutive frames that share acoustic similarities, the weak attention suppress method suppresses the redundant information for attention.

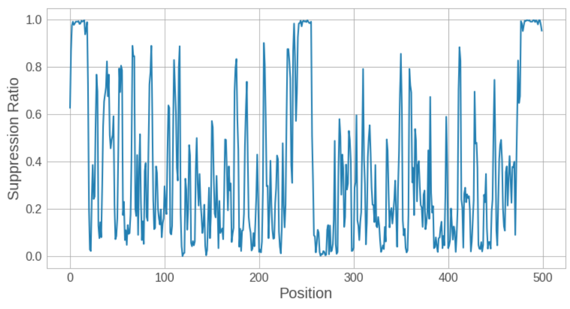
Comparing the Figures 1(b), 1(c) and 1(d) vertically, there are less fluctuations with higher layer. This observation may suggest that the lower transformer's self-attention can capture the small difference for consecutive frames while the higher layer captures the salient invariant features for consecutive frames.

Figure 2 gives the weak-attention suppression analysis at different positions from different transformer layers. For a specific position i, we average the portion of the suppressed attention over multiple heads and different utterances in the dev-clean dataset. We represent the process as follows:

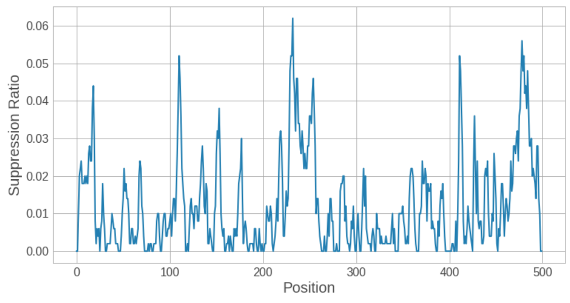
$$f_i(j) = \frac{\sum_{n=1}^{n=N} \sum_{k=1}^{k=H} s_{i,i+j}^{k,n}}{NH} \tag{9}$$



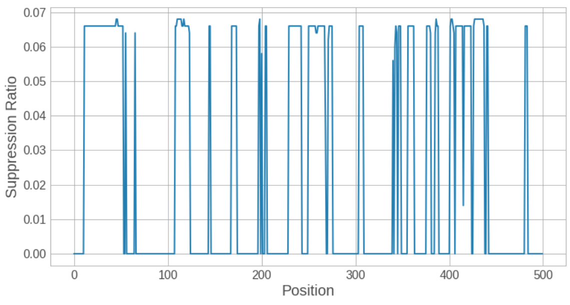
(a) wav



(b) layer 1



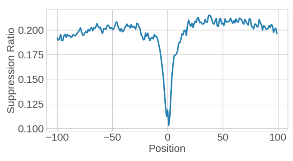
(c) layer 6



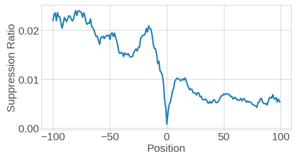
(d) layer 12

Figure 1: *Attention suppression analysis for one utterance in dev-clean using ‘vggTrf-12-WAS‘ in Table 1. (a) gives the audio wave of the utterance. (b), (c), (d) represent the average attention suppression portion for different positions in the input sequence at the 1-st, 6-th, 12-th transformer layer, respectively.*

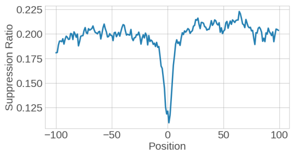
where $s_{i,j}^{k,n}$ means whether to suppress the attention between i-th position in *query* with $(i+j)$-th position in *key* from k-th head in utterance n. N is the number of utterances in dev-clean data and H the number heads in multi-head self-attention. The function value $f_i(j)$ indicates the weakness of the attention at the $i+j$-th position in the *key* for i-th position in *query*. For



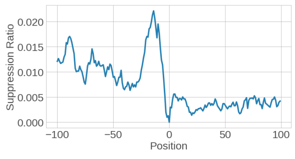
(a) position 200 layer 1



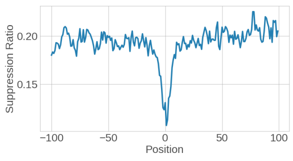
(b) position 200 layer 12



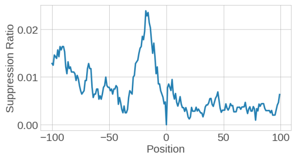
(c) position 400 layer 1



(d) position 400 layer 12



(e) position 600 layer 1



(f) position 600 layer 12

Figure 2: *Attention suppression analysis over the dev-clean using ‘vggTrf-12-WAS‘ in Table 1. Each sub-figure gives the average portion of attention suppressed at a specific position in a specific layer within the window size 200. We take the average over different heads in multi-head self-attention and different utterances over the dev-clean dataset.*

easy illustrating, we only draw the curve within window size 200 with 100 left and 100 right contexts, i.e., $j \in [-100, 100]$.

Figure 2 reveals the following three phenomenons. Firstly, the left column in the figure shows that more attention gets suppressed in the lower layer when the attention distance gets further. The deep valley of attention suppression happens in the context window of -5 to 5, i.e., 100 ms left context and 100 ms right context. The right column shows that in the 12-th layer, more attention gets suppressed from the left context than from the right context. One common phenomenon in both columns is that there are roughly 10 times more attention gets suppressed in the 1-st layer than the 12th layer.

5. Conclusions

In this paper, we proposed a weak-attention suppression (WAS) method to address the sparse attention for speech recognition. The attention probability smaller than a dynamic estimated threshold was zeroed out by WAS. The analysis showed several exciting phenomena of the proposed method. In the lower transformer layer, the attention suppression happens for the silence and redundant acoustic units. Even using the same attention suppression schema, less attention got suppressed in the upper layer than the lower layer. The further the dependency distance was, the more attention got suppressed. In the top transformer layer, the WAS suppressed more attention from the left context than the right context. Experiments on LibriSpeech showed that the WAS improves the transformer-based model with different model sizes in both streaming and non-streaming scenarios. Especially for a streamable transformer-based acoustic model, the proposed WAS got relative word error rate reduction by 10.0% on `test-clean` and 5.2% on `test-other`.

6. References

[1] 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.

[2] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive Language Models beyond a Fixed-Length Context," pp. 2978–2988, 2019.

[3] J. Devlin, M.-W. Chang, K. Lee, and Others, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," 2019. [Online]. Available: http://arxiv.org/abs/1910.10683

[5] L. Dong, S. Xu, and B. Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *Proc. ICASSP*, 2018.

[6] S. Karita, X. Wang, S. Watanabe, T. Yoshimura, W. Zhang, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, and R. Yamamoto, "A Comparative Study on Transformer vs RNN in Speech Applications," *2019 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019 - Proceedings*, pp. 449–456, 2019.

[7] M. Sperber, J. Niehues, G. Neubig, and Others, "Self-attentional acoustic models," *arXiv preprint arXiv:1803.09519*, 2018.

[8] S. Zhou, L. Dong, S. Xu, and B. Xu, "Syllable-based sequence-to-sequence speech recognition with the transformer in mandarin Chinese," *arXiv preprint arXiv:1804.10752*, 2018.

[9] 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," 2020. [Online]. Available: http://arxiv.org/abs/2002.02562

[10] C.-F. Yeh, J. Mahadeokar, K. Kalgaonkar, Y. Wang, D. Le, M. Jain, K. Schubert, C. Fuegen, and M. L. Seltzer, "Transformer-Transducer: End-to-End Speech Recognition with Self-Attention," 2019. [Online]. Available: http://arxiv.org/abs/1910.12977

[11] D. Povey, H. Hadian, P. Ghahremani, and Others, "A time-restricted self-attention layer for asr," in *Proc. ICASSP*, 2018, pp. 5874–5878.

[12] Y. Wang, A. Mohamed, D. Le, C. Liu, A. Xiao, J. Mahadeokar, H. Huang, A. Tjandra, X. Zhang, F. Zhang, C. Fuegen, G. Zweig, and M. L. Seltzer, "Transformer-based Acoustic Modeling for Hybrid Speech Recognition," 2019. [Online]. Available: http://arxiv.org/abs/1910.09799

[13] J. Salazar, K. Kirchhoff, and Z. Huang, "Self-Attention Networks for Connectionist Temporal Classification in Speech Recognition," in *Proceedings of ICASSP*, 2019. [Online]. Available: http://arxiv.org/abs/1901.10055

[14] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Processings of ICASSP*, 2015.

[15] C. Wu, Y. Wang, Y. Shi, C.-F. Yeh, and F. Zhang, "Streaming Transformer-based Acoustic Modeling Using Self-attention with Augmented Memory," in *Submitted to Interspeech 2020*, 2020.

[16] A. Mohamed, D. Okhonko, and L. Zettlemoyer, "Transformers with convolutional context for ASR," *arXiv preprint arXiv:1904.11660*, 2019.

[17] A. Tjandra, C. Liu, F. Zhang, and Others, "Deja-vu: Double Feature Presentation and Iterated loss in Deep Transformer Networks," *To appear ICASSP*, 2020.

[18] N. Kitaev, Ł. Kaiser, and A. Levskaya, "Reformer: The Efficient Transformer," 2020. [Online]. Available: http://arxiv.org/abs/2001.04451

[19] A. F. Martins and R. F. Astudillo, "From softmax to sparsemax: A sparse model of attention and multi-label classification," *33rd International Conference on Machine Learning, ICML 2016*, vol. 4, pp. 2432–2443, 2016.

[20] G. M. Correia, V. Niculae, and A. F. T. Martins, "Adaptively Sparse Transformers," pp. 2174–2184, 2019.

[21] D. Le, X. Zhang, W. Zheng, and Others, "From Senones to Chenones: Tied Context-Dependent Graphemes for Hybrid Speech Recognition," *2019 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019 - Proceedings*, 2019.

[22] D. Povey, A. Ghoshal, G. Boulianne *et al.*, "The kaldi speech recognition toolkit," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2011.

[23] T. Ko, V. Peddinti, D. Povey, and Others, "Audio augmentation for speech recognition," in *Proc. Interspeech*, 2015.

[24] D. S. Park, W. Chan, Y. Zhang, and Others, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.