Multi-View Frequency-Attention Alternative to CNN Frontends for Automatic Speech Recognition

Belen Alastruey[1], *Lukas Drude*[2], *Jahn Heymann*[2], *Simon Wiesler*[2]

[1]TALP Research Center, Universitat Politècnica de Catalunya, Barcelona, Spain
[2]Amazon Alexa, Germany

belen.alastruey@upc.edu, {drude, jahheyma, wiesler}@amazon.com

Abstract

Convolutional frontends are a typical choice for Transformer-based Automatic Speech Recognition (ASR) to preprocess the spectrogram, reduce its sequence length, and combine local information in time and frequency similarly. However, the width and height of an audio spectrogram denote different information, e.g., due to reverberation as well as the articulatory system, the time axis has a clear left-to-right dependency. On the contrary, vowels and consonants demonstrate very different patterns and occupy almost disjoint frequency ranges. Therefore, we hypothesize, global attention over frequencies is beneficial over local convolution. We obtain $2.4\,\%$ relative word error rate reduction (rWERR) on a production scale Conformer transducer replacing its Convolutional Neural Network (CNN) frontend by the proposed F-Attention module on Alexa traffic. To demonstrate generalizability, we validate this on public LibriSpeech data with an Long Short Term Memory (LSTM)-based Listen Attend and Spell (LAS) architecture obtaining $4.6\,\%$ rWERR and demonstrate robustness to (simulated) noisy conditions.

Index Terms: speech recognition, attention, robustness

1. Introduction

The advent of Transformer-based models [1] has surpassed the barriers of text. In ASR the Transformer typically operates on audio features like the mel-spectrogram [2]. These features provide longer input sequences compared to their text counterparts, and benefit from a frontend before the Transformer-based encoder [3, 4]. This frontend preprocesses the features and reduces their sequence length. The frontend of choice is often a CNN [3, 5], employing a stride for the sequence length compression. However, these layers were originally designed for image processing [6], and although direct use has been shown to work well, it might be sub-optimal due to the following reasons. Time and frequency are handled the same way although they contain different information. Furthermore, although speech information is usually concentrated in lower frequencies, the model is forced to process all frequency ranges equally. And finally, since attention is used just on tokens obtained after splitting the features time-wise, the extraction of frequency interactions is limited by the usually small kernel size in the CNN frontend.

In the past, frequency LSTMs (F-LSTMs) have been proposed as a frontend to LSTM-based acoustic models to better capture frequency dependencies. The frontend operates on time steps independently and scans the different frequency bins. Then, the hidden states of the F-LSTM are concatenated and used as input for the acoustic model. An extension of this method, the multiview frequency LSTM (mvF-LSTM) [7], uses

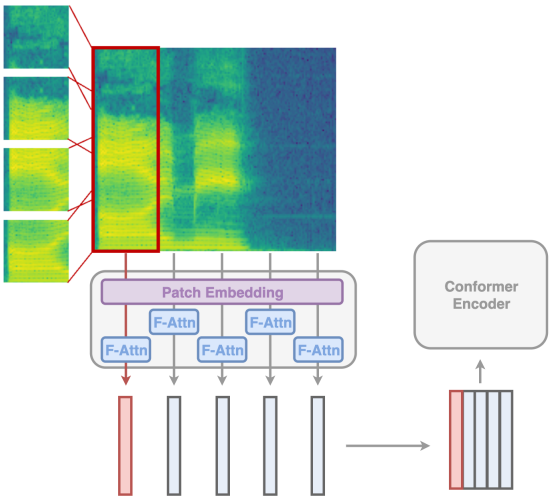

Figure 1: *Overview of the proposed F-Attention frontend. The spectrogram is split into overlapping patches. Then, we use attention to extract interactions between patches on the same time range but on different frequency bins. We concatenate the outputs time-wise to obtain the input of the Conformer encoder.*

many F-LSTMs with different window sizes and strides (referred to as "views") for each time step. Then, the output of each F-LSTM is combined using a projection layer and the resulting vector is the input of the acoustic model.

Although F-LSTM could seem an adequate alternative to solve the issues of CNN frontends, these networks are not suitable either to preprocess a spectrogram when using a Transformer-based model. The concatenation of the different hidden states of the F-LSTM on each time step causes an increase of the embedding dimension by a factor of 8. As a consequence, the number of parameters in the projection matrices in the attention layers of the Transformer-based encoder would increase remarkably. MvF-LSTMs bypass this problem through the use of a projection layer that is intended to merge the different views, and that can simultaneously be used to compress the embedding dimension. However, the number of parameters on this linear layer scales drastically when adding views, and would not be comparable to a CNN frontend. But most importantly, neither F-LSTMs nor mvF-LSTMs compress the sequence length. This can be a problem when regarding complexity, since the Transformer's attention matrix computational cost scales with $\mathcal{O}(n^2)$, where n is the sequence length.

Other frontend alternatives have been inspired by vision Transformers [8], such as the Audio Spectrogram Transformer for audio classification [4]. Then, the patches are embedded and reordered to form a sequence that is used as the input of a BERT-

like classifier [9]. Thanks to the patches rearrangement for the input sequence formation, this frontend allows the Transformer-based encoder to analyze the interaction between different frequencies (patches corresponding to the same time range but to different frequency bins will be in separate tokens in the final sequence). However, this approach does not reduce the sequence length either, and in fact, the final sequence is even longer than the original one. Furthermore, the sequence obtained after the patches rearrangement is interleaved which may be problematic for a sequence-to-sequence task.

In this work, we present a new frontend based on two main ideas: (1) patch extraction, inspired by vision Transformers, and (2) the analysis of frequency interactions, inspired by F-LSTMs and mvF-LSTMs. We propose the F-Attention frontend, which extracts patches out of a spectrogram and then uses attention to extract the relationships between patches on a same time range but on different frequency bins. This frontend aims to improve the way speech features are processed, by being able to focus on relevant frequency ranges. Our proposed frontend obtains performance improvements with respect to the CNN baseline. We show how F-Attention is able to dismiss noise and focus on relevant frequency ranges using an interpretability method and evaluate noise robustness.

2. mvF-Attention Frontend

The proposed frontend has two main steps: (1) patch extraction, with one or multiple patch sizes ("views"), and (2) F-Attention, with one or multiple layers of self-attention as illustrated in Figures 1 and 2. The layers of each view operate independently and are only merged at the end via a pooling operation.

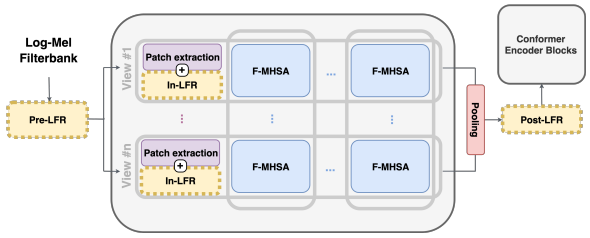

Figure 2: *mvF-Attention frontend with configurable number of views and layers. The downsampling positions (Pre-LFR, In-LFR, Post-LFR) are optional and compared in Section 3.1.1.*

To let the model learn different relationships, we extract patches of the spectrogram for each view using different patch sizes. In each view we use multi-head self-attention to analyse the interactions between patches that correspond to the same time range but to different frequency bins. We do not share parameters neither across the self-attention blocks inside a view nor within a layer (in different views) to ensure that each self-attention analyses different patterns. Following the Transformer architecture [1], each multihead self-attention (MHSA) is followed by a residual connection and layer normalization.

To merge the outputs of the different views, mvF-LSTMs concatenate the output of each view and use a linear layer to project it to the dimension of the encoder. However, we notice that that leads to an escalation of the number of the parameters in the linear layer when stacking many views. To bypass this problem, we decide to merge the different views using a pooling layer in the patch embedding dimension. Therefore, the size of the output of the frontend is kept constant regardless of the number of views, maintaining a reasonable number of parameters in the linear layer.

3. Experimental Results

This section describes the implementation details of our models, the datasets and training setup. We conduct experiments on internal data with a conformer transducer architecture first and validate findings with a LAS architecture on a public dataset subsequently.

3.1. Voice-controlled far-field experiments

We train a Conformer Transducer model [3] with 73.5 M parameters on about 50 000 h of de-identified English speech from voice-controlled far-field devices. In the Conformer Transducer the LSTM encoder of an RNN-T model [10, 11, 12] is replaced with a Conformer encoder [3]. The baseline frontend, with a total of 3.3 M parameters, has 2 CNN layers of kernel 3×3, stride 2×2 and an embedding dimension of 128. These are followed by a linear layer that projects the tensor to the encoder embedding dimension. F-Attention frontends, with a number of parameters from 3.2 M to 3.5 M depending on the variant, have a patch embedding layer for each view, with different patch sizes and a stride of 4×4 so that the compression factor is equal to the baseline[1]. In all setups, we selected 7×7 patches when using one view and extended to 7×7, 14×14 and 3×3, 7×7, 14×14, 28×28 when using two and four views, respectively. The patch embedding dimension is 128. Each of the self-attention layers in the frontend has 8 heads and a dimension of 128. The Conformer encoder comprises 12 Conformer blocks with a total of 50.9 M parameters. Each Conformer block has 8 self-attention heads, and an encoder dimension of 512. The predictor network consists of a 2×768 LSTM with a total of 15.2 M parameters makes up the prediction network. A feed-forward layer with 512 units and *tanh* activation makes up the first layer of the joint network, which is followed by a *softmax* layer with an output vocabulary size of 4 001 word pieces.

We train each model for 500 000 steps with the Adam optimizer [13], a learning rate of 0.0001, a linear warm-up of 5 k iterations, and an exponential decay. Additionally, we group training examples by sequence length with bucket boundaries $\{300, 600\}$ and corresponding bucketing batch sizes of $[16, 8, 2]$. Unless otherwise stated, the acoustic features are 64-dimensional Log FilterBank Energy (LFBE) features with a window size of 25 ms and a frame shift of 10 ms. As an augmentation method, we use a variant of SpecAugment [14], as proposed in [15]. For the evaluation, the top 5 best checkpoints out of the last 10 are selected based on validation dataset word error rate (WER). Then, the weights of the top 5 models are averaged and used to obtain the final test scores.

Our experiments follow four main directions: (1) studying the position of Low Frame Rate (LFR) conversion with respect to the proposed frontend, (2) analyzing the proposed frontend with variations in the number of views or attention layers, (3) studying the generalizability to different speech features, and (4) exploring the ability of the F-Attention to ignore noise and focus on relevant frequency ranges.

[1]Note that a patch size of 7×7 and a stride of 4×4 replicates the behaviour of the two convolutional layers in the baseline.

3.1.1. *Optimal position of LFR conversion*

In the baseline model, speech features are stacked and downsampled by a factor of 3, corresponding to an encoder frame rate of 30 ms. Although this is done before the frontend, we hypothesize that F-Attention works optimally on less-processed features. Therefore, we consider three different approaches, shown in yellow in Figure 2. Pre-LFR is the standard approach, features are stacked and downsampled before the frontend, as in the baseline. In-LFR conversion consists of a modification of the strides and and patch shapes in the patch extraction layer so that the downsampling is done simultaneously. Finally, Post-LFR conversion consists of the same transformation used in Pre-LFR conversion, but it is employed after the frontend. For any of the three approaches, the final encoder frame rate is 30 ms.

Table 1 summarizes this initial study. For a F-Attention frontend with one view and one layer we observe that Pre-LFR as well as In-LFR degrade over the baseline. Only Post-LFR is on par with the baseline. Observing similar trends on the development partition we use the Post-LFR setting going forward and do not revisit this decision for more views or more layers.

Frontend	LFR Position	rWERR (%, ↑)
Baseline	Pre-LFR	**0.0**
F-Attention	Pre-LFR	−1.2
F-Attention	In-LFR	−0.5
F-Attention	Post-LFR	**0.0**

Table 1: *Study of different LFR conversion strategies combined with our frontend in terms of rWERR. ↑ : higher is better.*

3.1.2. *Optimal configuration of layers and views*

To evaluate the performance of our frontend and find the optimal configuration, we explore different combinations on the number of views and layers, keeping the number of parameters close to the baseline. While the baseline model has 73.5 M of which 3.3 M are the frontend, all our proposed frontends have between 3.2 M and 3.5 M parameters. To study the consistency of our results, we train two variants of the models for each views/layers combination: Conformer transducer with joint network and without it (with a single linear projection).

The results shown in Table 2 indicate consistent gains when using the F-Attention frontend with or without the joint network. Comparing the experiments with one layer and multiple views against those with one view but multiple layers, we can conclude that adding views can have a better impact on the model performance. Furthermore, observing the experiment with two views and two layers, we obtain the best performance obtained by any of our models. However, we decide not to try more configurations with simultaneously multiple views and layers due to the additional increase in the number of parameters and GPU memory requirements.

3.1.3. *Behaviour on Different Speech Features*

To understand how well the F-Attention frontend generalizes to other features, we perform a brief comparison of F-Attention frontends (choosing the two more comparable in number of parameters) on 256-dimensional Log Short Time Fourier Transform (LSTFT) features instead of 64-dimensional LFBE features used in previous experiments.

Layers/ views	Params (M)	rWERR(%, ↑)	
		w/o Joint Net	w/ Joint Net
Baseline	3.3	0.0*	0.0**
1 / 1	3.2	0.0	−0.4
1 / 2	3.3	**+2.1**	+1.3
1 / 4	3.5	+1.9	+2.2
2 / 1	3.3	+1.4	+1.1
4 / 1	3.4	+0.9	+0.7
2 / 2	3.4	+1.2	**+2.4**

Table 2: *Study of different number of layers and views of the F-Attention frontend. For every experiment we compare the number of parameters of the frontend and the performance of the model measured on rWERR. Each model is compared to the respective baseline with or without joint network. ↑ : higher is better. *: Corresponds to last row in Table 1. **: +5.8 wrt. baseline without joint network.*

We observe that a model with F-Attention frontend with 1 layer and 2 views improves by 1 % relative over the corresponding baseline with convolutional frontend whereas an F-Attention frontend with 2 layers and 1 view just leads to 0.4 % rWERR. We conclude that the approach generalizes to this feature variant but concentrate on LFBE going forward.

3.2. Interpretability Analysis of the mvF-Attention

The use of F-Attention enables the model to focus on specific frequency ranges and dismiss the information on others. This ability can be useful on noisy backgrounds, where the frontend could be able to ignore the noisiest frequency ranges.

In this regard, aiming to understand if F-Attention is processing the spectrogram as expected, we perform an interpretability analysis on the frontend. To do so, we employ ALTI [16], a token attribution method that summarizes information from the self-attention layer (attention weights, values and output projection), the residual connection and the layer normalization. Using this method, we obtain a contributions plot (Figure 4, C is defined as in Eq. 10 in [16]) that shows how every input token (columns) contributes to each output token (rows) when doing a forward pass of the embedded patches through the F-Attention layer. For the sake of simplicity, we decide to analyse our most simple model, the F-Attention frontend with one view and one layer. The F-Attention layer operates inside specific time ranges, extracting interactions frequency-wise. We analyse the behaviour of the frontend on a specific time range shown in Figure 3.

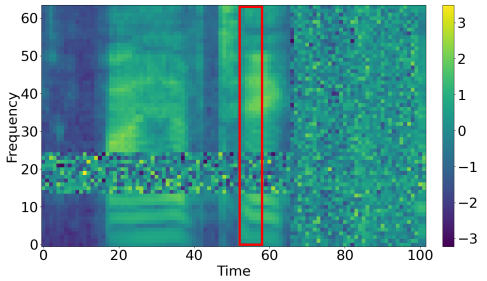

Figure 3: *Analysis of a spectrogram taken from the public LibriSpeech corpus. Time range to study highlighted red.*

Analysing the results in Figure 4, we can see how those in-

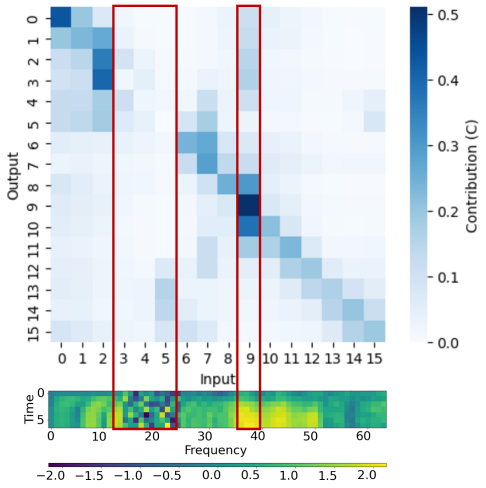

Figure 4: *Zoom of the analyzed time range and token attributions. We see how patches that contain noise added by Spec Augment (from 3rd to 5th) don't contribute to any output token.*

put tokens that correspond to patches that contain noise added by SpecAugment (from 3rd to 5th) have a negligible contribution to every output token. Additionally, we see that patch number 9, which represents a highlighted frequency range has a big impact to many output tokens. Therefore, the F-Attention layer is able to detect the most relevant frequency ranges and ignores those that are not useful for the transcription.

3.3. Verification on public data

To verify the efficacy of the proposed front-end on public data, we also evaluate the approach on LibriSpeech. For this dataset, the model consists of six-layer bi-directional LSTM [17] with 1,024 units encoder and a two-layer uni-directional LSTM decoder with a single head content-based attention mechanism [18]. Overall the model has 104M trainable parameters. The model has been trained with Adam optimizer [19]. We use a SentencePiece [20] unigram word-piece model with 4000 tokens. SpecAugment data augmentation method with LD policy [21] was used throughout model training. The model was trained with weight noise [22], which was added to all encoder trainable parameters by sampling noise from a normal distribution with a standard deviation of 0.075 starting from 15k training steps. We use uniform label smoothing [23] which distributes a probability mass of 0.1 to non-ground truth tokens.

Table 3 summarizes results for different combinations of number of layers, number of views and patch embedding size E. In line with aforementioned joint network results on internal data the configuration with two layers and two views performs best, leading to $4.6\,\%$ rWERR on the *test other* partition. We further observe that reducing the patch embedding size E reduces the number of parameters but shows mixed results when comparing the *test clean* and the *test other* partition. In all cases, adding more views leads to better performance on *test other* with mixed results on *test clean*.

3.3.1. Robustness to Noise Addition

The results shown in section 3.2 show that F-Attention is able to ignore noise and focus on relevant frequency ranges. To extend this study we evaluate our best LibriSpeech models in terms of performance and comparability to the baseline on the same test set but with different noise conditions. We decided to add

Layers/ views	E	Params (M)	WER(%, ↓)		rWERR(%, ↑)	
			Clean	Other	Clean	Other
Baseline		104.7	3.1	8.2	+0.0	+0.0
1 / 1	128	105.8	3.1	8.1	−0.6	+1.3
2 / 1	128	105.9	3.1	8.1	−0.1	+1.0
2 / 2	128	106.0	**3.0**	**7.8**	**+3.9**	**+4.6**
1 / 1	64	105.2	3.1	8.3	+0.4	−1.6
1 / 2	64	105.3	**3.0**	8.2	+1.7	−0.5
1 / 4	64	105.3	3.1	8.2	−1.7	+0.1

Table 3: *LibriSpeech test results. Comparison of F-Attention frontends against a convolutional frontend using an LSTM-based LAS model with a patch embedding size E.*

internally-recorded babble background noise to utterances of the test clean partition of LibriSpeech. Figure 5 shows rWERR for different different signal to noise ratio (SNR) conditions. We observe that the evaluated configurations improve substantially over the baseline in very noisy conditions, i.e. SNR < 0. This effect is most pronounced for the 1-layer 1-view F-attention configuration. Interestingly, all systems improve over the baseline at about $10\,\mathrm{dB}$ SNR. A possible explanation is, that a certain noise floor covers up artifacts of the processing pipeline or that slightly noisy conditions better reflect the training conditions.

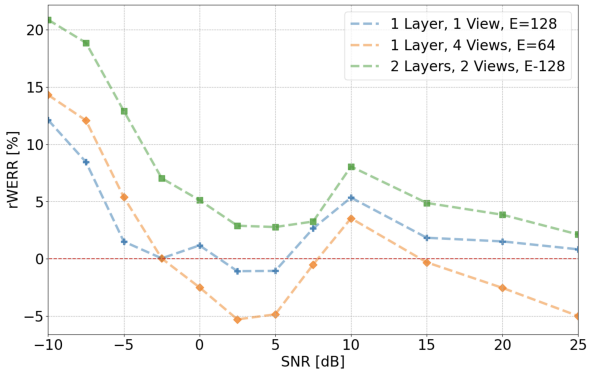

Figure 5: *Robustness of different front-ends in comparison to the baseline for different SNR conditions on LibriSpeech with internally-recorded bubble background noise. All results are rWERR. Higher is better.*

4. Conclusions

In this paper we have proposed frequency attention as a new Transformer frontend that processes speech using self-attention over frequency bins. Our proposed frontend leads to $2.4\,\%$ rWERR compared to a convolutional frontend on large in-house data with a Conformer-Transducer model. We demonstrated that these gains transfer to public data (here LibriSpeech) and a different ASR architecture (here LAS). On the LibriSpeech *test other* partition, the proposed frontend obtains a $4.6\,\%$ rWERR. We demonstrated that these findings hold for LSTFT and LFBE features. Using an interpretability method we have been able to see how the frontend discards noise and focuses on relevant frequency ranges. This insight was further corroborated by a analyzing the F-attention performance in a wide range of SNRs with relative gains over $20\,\%$ in very noisy conditions.

5. References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[2] 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)*, 2018, pp. 5884–5888.

[3] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented Transformer for Speech Recognition," in *Proc. Interspeech 2020*, 2020, pp. 5036–5040.

[4] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *Proc. Interspeech 2021*, 2021, pp. 571–575.

[5] C. Wang, Y. Tang, X. Ma, A. Wu, D. Okhonko, and J. Pino, "Fairseq S2T: Fast speech-to-text modeling with fairseq," in *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*. Suzhou, China: Association for Computational Linguistics, Dec. 2020, pp. 33–39. [Online]. Available: https://aclanthology.org/2020.aacl-demo.6

[6] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[7] M. Van Segbroeck, H. Mallidih, B. King, I.-F. Chen, G. Chadha, and R. Maas, "Multi-view frequency lstm: An efficient frontend for automatic speech recognition," 2020. [Online]. Available: https://arxiv.org/abs/2007.00131

[8] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[10] A. Graves, "Sequence transduction with recurrent neural networks," in *Proceedings Representation Learning Workshop on International Conference on Machine Learning (ICML)*, Edinburgh, Scotland, 2012.

[11] C. Zhang, B. Li, Z. Lu, T. N. Sainath, and S.-y. Chang, "Improving the fusion of acoustic and text representations in rnn-t," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 8117–8121.

[12] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving rnn transducer modeling for end-to-end speech recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 114–121.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[14] 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.

[15] D. S. Park, Y. Zhang, C.-C. Chiu, Y. Chen, B. Li, W. Chan, Q. V. Le, and Y. Wu, "Specaugment on large scale datasets," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6879–6883.

[16] J. Ferrando, G. I. Gàllego, and M. R. Costa-jussà, "Measuring the mixing of contextual information in the transformer," in *Findings of the Association for Computational Linguistics: EMNLP 2022*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, p. 8698–8714. [Online]. Available: https://aclanthology.org/2022.emnlp-main.595

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[18] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho *et al.*, "Attention-based models for speech recognition," in *NIPS*, 2015.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.

[20] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," 2018.

[21] D. S. Park, W. Chan, Y. Zhang, C. Chiu *et al.*, "Specaugment: A simple data augmentation method for automatic speech recognition," in *INTERSPEECH*, 2019.

[22] A. Graves, "Practical variational inference for neural networks," in *NIPS*, 2011.

[23] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *INTERSPEECH*, 2017.