



Stochastic Attention Head Removal: A Simple and Effective Method for Improving Transformer Based ASR Models

Shucong Zhang, Erfan Loweimi, Peter Bell, Steve Renals

Centre for Speech Technology Research, University of Edinburgh, Edinburgh, UK

s1603602@sms.ed.ac.uk, {e.loweimi, peter.bell, s.renals}@ed.ac.uk

Abstract

Recently, Transformer based models have shown competitive automatic speech recognition (ASR) performance. One key factor in the success of these models is the multi-head attention mechanism. However, for trained models, we have previously observed that many attention matrices are close to diagonal, indicating the redundancy of the corresponding attention heads. We have also found that some architectures with reduced numbers of attention heads have better performance. Since the search for the best structure is time prohibitive, we propose to randomly remove attention heads during training and keep all attention heads at test time, thus the final model is an ensemble of models with different architectures. The proposed method also forces each head independently learn the most useful patterns. We apply the proposed method to train Transformer based and Convolution-augmented Transformer (Conformer) based ASR models. Our method gives consistent performance gains over strong baselines on the Wall Street Journal, AISHELL, Switchboard and AMI datasets. To the best of our knowledge, we have achieved state-of-the-art end-to-end Transformer based model performance on Switchboard and AMI.

Index Terms: speech recognition, end-to-end, self-attention, Transformer, stochastic attention head removal

1. Introduction

In self-attention based models such as Transformers [1], each self-attention layer uses multi-head attention to capture a set of different inputs representations, and have given competitive ASR results [2–10]. However, we previously observed that not all of the attention heads are useful [11]. We found that in trained Transformer ASR models, the attention matrices of some attention heads are close to the identity matrices, indicating that the attention mechanism is just an identity mapping. Thus, the input sequence (after a linear transformation) can be directly used as the output of the corresponding attention head.

This observation leads to a question: if there are redundant attention heads in trained Transformer models, then when we train a model from scratch, can we remove some attention heads without harming the model performance? In our experiments, we found some architectures with reduced numbers of attention heads offer better accuracies. However, searching for the best structure is time consuming. To address this, we propose to randomly remove attention heads during training, and keep all the attention heads at test time. Through this method, the trained model is an ensemble of different architectures.

Our method also enables the networks to better utilize the representation power of the multi-head attention architecture. The redundancy of attention heads in the baselines indicates some heads learn the most crucial patterns while other heads merely extract inessential features which can be disregarded (this is also observed in [12]). In contrast, our method forces

all attention heads to learn crucial features – since heads are randomly removed during training, the model cannot only rely on some specific heads for extracting key features. Thus, the proposed method forces each head to learn useful information and enables the model to better use the representation power of the multi-head attention structure. Our experimental results and analysis on attention matrices support our arguments.

We employ the proposed method to train Transformer and Conformer [10] ASR models. Our method offers consistent accuracy improvements on datasets of different ASR scenarios, including Wall Street Journal (WSJ) [13], AISHELL [14], Switchboard (SWBD) [15] and AMI [16]. On SWBD and AMI, to the best of our knowledge, we have achieved state-of-the-art (SOTA) performance for end-to-end Transformer based models.

2. Related Work

At first sight, the proposed method appears to be similar to dropout [17]; however the objectives of these two methods are different. Dropout randomly drops units in neural networks to prevent the co-adaptation between these units. In this work, we randomly remove entire attention heads, motivated by the observation that some architectures with reduced numbers of attention heads can lead to better performance, but avoiding the need for a time-consuming search for the best architecture. Instead, we effectively train an ensemble of different structures. Furthermore, we observe our method forces the attention heads which are redundant in the baseline systems to learn useful patterns. We also have employed dropout in our experiments and found it is complementary to our method.

Previous works have proposed to use a development set [18] or a trainable hard gate head [12] to drop attention heads. These methods remove attention heads in a static manner – once an attention head is removed during training, it will never be present in the structure. Also, little performance gain is obtained by these methods. Our proposed method removes attention heads randomly and consistently yields better accuracies.

To alleviate the gradient vanishing problem and build very deep networks, previous works have proposed to drop self-attention layers randomly or following a schedule [3, 19]. Rather than dropping self-attention layers, our method removes individual attention heads randomly. The motivation of our method also differs from the previous works. Thus, the proposed method is complementary with these previous works.

Peng et al [20] proposed to train subsets that contain $h - 1$ elements of the h attention heads. During testing, the output is a linear combination of the outputs of these subsets. In our work, the attention heads are randomly removed. Therefore, the number of trainable attention heads changes dynamically.

Zhou et al [21] proposed a similar method of training self-attention models for natural language processing (NLP). However, the related work does not analyse where the benefits of

such methods come from, whilst in contrast, we independently derive our method based on our previous work which studies the usefulness of the attention heads [11]. Additionally, in this work, our detailed analysis of the proposed method reveals why dynamically removing attention heads is helpful. Furthermore, the previous work relates to NLP while our method focus on models which are specifically designed for ASR.

3. Transformer Based Models

In this section, we briefly review Transformer [1] and Conformer [10]. The basic building block for Transformer based models is self-attention [1]. A self-attention layer takes a 3-tuple of sequences as its input. Each self-attention layer has a multi-head attention (MHA) component and a feed-forward component. Each attention head A_i in the MHA uses a linear transformation to map the input sequences to a Query/Key/Value triplet, which is denoted as $(\mathbf{Q}, \mathbf{K}, \mathbf{V})$. Then, it uses the dot product to compute the similarity between each element of the \mathbf{K} sequence and each element of the \mathbf{Q} sequence. Using these similarities as weights, each element of the output sequence of the attention head is the weighted sum of the \mathbf{V} sequence. Thus, an attention head A_i can be described as:

$$A_i(\mathbf{X}^{\mathbf{Q}}, \mathbf{X}^{\mathbf{K}}, \mathbf{X}^{\mathbf{V}}) = \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d^{\mathbf{K}}}}\right) \mathbf{V}_i \quad (1)$$

$$(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) = (\mathbf{X}^{\mathbf{Q}} \mathbf{W}_i^{\mathbf{Q}}, \mathbf{X}^{\mathbf{K}} \mathbf{W}_i^{\mathbf{K}}, \mathbf{X}^{\mathbf{V}} \mathbf{W}_i^{\mathbf{V}}) \quad (2)$$

where $\mathbf{X}^{\mathbf{Q}} \in \mathbb{R}^{n \times d^{\mathbf{M}}}$, $\mathbf{X}^{\mathbf{K}}, \mathbf{X}^{\mathbf{V}} \in \mathbb{R}^{m \times d^{\mathbf{M}}}$ are inputs and m, n denote the lengths of the input sequences; $\mathbf{W}_i^{\mathbf{Q}}, \mathbf{W}_i^{\mathbf{K}} \in \mathbb{R}^{d^{\mathbf{M}} \times d^{\mathbf{K}}}$ and $\mathbf{W}_i^{\mathbf{V}} \in \mathbb{R}^{d^{\mathbf{M}} \times d^{\mathbf{V}}}$ are trainable matrices.

The multi-head attention MHA is a linear combination of the outputs of the h attention heads (A_1, A_2, \dots, A_h) , which can be described as:

$$\text{MHA}(\mathbf{X}^{\mathbf{Q}}, \mathbf{X}^{\mathbf{K}}, \mathbf{X}^{\mathbf{V}}) = (A_1, A_2, \dots, A_h) \mathbf{U}^{\mathbf{H}}, \quad (3)$$

where $\mathbf{U}^{\mathbf{H}}$ is a trainable matrix and $\mathbf{U}^{\mathbf{H}} \in \mathbb{R}^{d^{\mathbf{H}} \times d^{\mathbf{M}}}$, $d^{\mathbf{H}} = h \times d^{\mathbf{V}}$. A residual connection links the the output of the MHA and the query sequence $\mathbf{X}^{\mathbf{Q}}$:

$$\mathbf{X}' = \mathbf{X}^{\mathbf{Q}} + \text{MHA}(\mathbf{X}^{\mathbf{Q}}, \mathbf{X}^{\mathbf{K}}, \mathbf{X}^{\mathbf{V}}) \quad (4)$$

Finally, there is a feed-forward component upon the MHA:

$$\mathbf{Y} = \mathbf{X}' + \text{ReLU}(\mathbf{X}' \mathbf{S} + \mathbf{b}) \mathbf{Z} + \mathbf{r} \quad (5)$$

where $\mathbf{S} \in \mathbb{R}^{d^{\mathbf{M}} \times d^{\mathbf{FF}}}$, $\mathbf{Z} \in \mathbb{R}^{d^{\mathbf{FF}} \times d^{\mathbf{M}}}$, $\mathbf{b} \in \mathbb{R}^{d^{\mathbf{FF}}}$ and $\mathbf{r} \in \mathbb{R}^{d^{\mathbf{M}}}$ are trainable matrices and vectors.

Transformer is a self-attention based model, which has a self-attention encoder to encode the input sequences and a self-attention decoder to decode the encoded sequences. For each self-attention layer in the encoder, $\mathbf{X}_e^{\mathbf{Q}} = \mathbf{X}_e^{\mathbf{K}} = \mathbf{X}_e^{\mathbf{V}}$ and each sequence is the speech signal or its hidden representation from the previous layer. Each self-attention layer in the decoder has two MHA components and one feed-forward component. The first MHA attends the previous output tokens or the outputs of the previous layer. Then, the second MHA looks at the encoded speech signal, $\mathbf{X}_s^{\mathbf{K}} = \mathbf{X}_s^{\mathbf{V}}$ are the final hidden representations of the input speech sequence, and $\mathbf{X}_s^{\mathbf{Q}}$ is the output of the previous MHA (with the residual connection).

Since self-attention encodes sequences through attention mechanisms, it can model dependencies within any range. However, previous works have shown it is less effective in capturing local information [11, 22]. Nevertheless, acoustic events

usually happen within short periods and thus local information is essential for ASR tasks. Since convolution neural networks (CNNs) are suitable for capturing local context, Gulati et al [10] propose Conformer, which augments self-attention layer encoders with CNNs. A Conformer layer can be described as:

$$\tilde{\mathbf{X}} = \mathbf{X} + \frac{1}{2} \text{FF}(\mathbf{X}) \quad (6)$$

$$\mathbf{X}' = \tilde{\mathbf{X}} + \text{MHA}(\tilde{\mathbf{X}}, \tilde{\mathbf{X}}, \tilde{\mathbf{X}}) \quad (7)$$

$$\mathbf{X}'' = \mathbf{X}' + \text{Conv}(\mathbf{X}') \quad (8)$$

$$\mathbf{Y} = \text{LayerNorm}(\mathbf{X}'' + \frac{1}{2} \text{FF}(\mathbf{X}'')) \quad (9)$$

where FF denotes the feed-forward component [10], Conv denotes the convolution component [10] and LayerNorm denotes layer normalisation [23]. The half-step feed-forward networks $\frac{1}{2} \text{FF}$ have resulted in better accuracies compared to the vanilla feed-forward network [10]. In the original Conformer model, long short-term memory (LSTM) [24] is used as the decoder. In this paper, we use ‘‘Conformer’’ to denote the **Conformer Encoder–Transformer Decoder** structure.

4. Stochastic Attention Head Removal

We introduce the stochastic attention head removal (SAHR) strategy in this section. In our previous work [11], we found in trained Transformer ASR models, there are heads which only generates nearly diagonal attention matrices. The high diagonality indicates these heads are merely identity mappings and thus they are redundant. We further investigate if removing the heads with high diagonality and retrain the model from scratch will impact the performance. We use WSJ as the dataset and the experimental setups are presented in Section 5. For the baseline model, following our previous work, we plot the heatmap of the averaged diagonality [11] on WSJ eval92 for each head in every encoder layer. Then, we remove the heads whose averaged diagonality are above a threshold and retrain the model. We also test removing all the heads in the uppermost encoder layer (near output), since all the heads in this layer have high diagonality. Figure 1 shows the heatmap and the structures with the corresponding head removal strategy.

Table 1 shows the character error rates (CERs) for each structure. We observe some architectures outperform other structures. However, the search for the best architecture is time-consuming. To address this, we propose to randomly remove attention heads during training and keep all heads at test time. The proposed method can be viewed as training different architectures and using the ensemble of these trained architectures as the final model at test time. When processing a training example, each attention head has a probability q of being removed, where q is set as a hyper-parameter. If a head is kept, then the output of that head is scaled by $\frac{1}{1-q}$:

$$A_i(\mathbf{X}^{\mathbf{Q}}, \mathbf{X}^{\mathbf{K}}, \mathbf{X}^{\mathbf{V}}) = \frac{1}{1-q} \times \text{softmax}\left(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d^{\mathbf{K}}}}\right) \mathbf{V}_i \quad (10)$$

At test time, the scale factor $\frac{1}{1-q}$ is removed and all the attention heads are always present. Thus, the expected output of the attention head is the same during training and testing. During training, if all the attention heads of a layer are removed for an utterance, then that layer becomes a feed-forward layer. The stochastic attention head removal strategy is applied for both the encoder and the decoder. Table 1 shows the preliminary results of our method. In Section 5, we further test applying the proposed method to train both Transformer and Conformer ASR models on other larger and more challenging datasets.

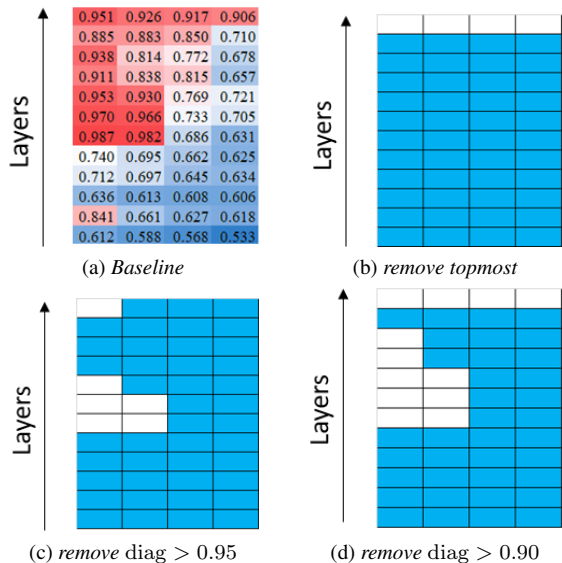


Figure 1: The tested encoder architectures. The baseline has 12 self-attention layers. Each non-white cell represents an attention head. Each white cell denotes there is no attention head. When all the attention heads of a layer are removed (four white cells), the feed-forward component of that layer is preserved. (a) heatmap of the averaged diagonality of each attention head. (b-d) structures where some attention heads are removed.

Table 1: CERs on WSJ for different removal strategies. The structures of the static removal methods are shown in Figure 1. #heads/ denotes the number of attention heads.

Removal Method	# Heads	eval 92/dev 93
None (baseline)	48	3.5/4.6
Remove topmost heads	44	3.4/4.5
Remove heads diag > 0.95	42	3.4/4.6
Remove heads diag > 0.90	36	3.5/4.7
Random removal with $q = \frac{6}{48}$	48	3.4/4.5

5. Experimental Results

5.1. Setup

We experiment on datasets of a variety of scenarios. Table 2 describes these datasets. We use Kaldi [25] for data preparation and feature extraction – 83-dim log-mel filterbank frames with pitch [26]. SpecAugment [27] is used on all datasets but WSJ. On all datasets, following the previous works [2, 4, 28], we employ the **12layer Encoder-6layer Decoder** architecture. The multi-head attention components of the self-attention layers have 4 attention heads and $d^V = d^K = 64$, $d^M = 256$. For the feed-forward module of the self-attention layers, $d^{FF} = 2048$. Below the encoder of the Transformer, there are two convolutional neural network layers with 256 channels, with a stride of 2 and a kernel size of 3, which map the dimension of the input sequence to d^M . The structure of the self-attention component of the Conformer is the same as the Transformer. The kernel size of the convolution component is either 15 or 31 (depends on the dataset). Input sequences to the encoder and the decoder are concatenated with sinusoidal positional encoding [1, 29]. Models are implemented using ESPnet [30] and PyTorch [31].

The training schedule follows [2]. Dropout rate 0.1 is used. The training lasts 100 epochs for WSJ and 50 epochs for other

Table 2: Datasets Description

datasets	language	hours	speech style
WSJ	EN	81	read
AISHELL	ZH	170	read
SWBD	EN	260	telephone conversation
AMI	EN	100	meeting

Table 3: WERs on SWBD for Transformer models.

Removal Probability	SWBD/Callhome (WER%)		
	seed 1	seed 2	seed 3
Transformer Models			
0/48 (baseline)	9.0/18.1	9.1/18.2	9.1/17.9
4/48	8.9/17.5	8.7/17.6	-
6/48	8.6/17.2	8.7/16.9	8.7/16.8
8/48	8.8/17.6	8.9/17.2	-
Transformer [4]			9.0/18.1
Very deep self-attention [3]			10.4/18.6
Multi-stride self-attention [5]			9.1/-

datasets. The averaged parameters of the last 10 epochs are used as the parameters of the final model [2]. Adam [32] is used as the optimizer. The output units for WSJ/AISHELL/AMI are characters and the output tokens for SWBD experiments are tokenized at word-level using Byte Pair Encoding (BPE) [33]. The batch size is 32. Label smoothing with smoothing weight 0.1 is used. Besides the loss from the Transformer’s decoder L^D , a connectionist temporal classification (CTC) [34] loss L^{CTC} is also applied to the Transformer/Conformer encoder [35]. Following [4], the final loss L for the model is:

$$L = (1 - \lambda)L^D + \lambda L^{CTC}. \quad (11)$$

where $\lambda = 0.2$ for SWBD and $\lambda = 0.3$ for other datasets. Following our preliminary study (Table 1), considering both the encoder and the decoder has 48 attention heads, we test removal probability values starting from $\frac{4}{48}$.

5.2. Results and discussion

We use WSJ to derive the proposed method and the experimental results are shown in Table 1. Table 3 shows the results of applying the proposed method to train Transformers on SWBD. We run the experiments with three different random seeds, and have obtained consistently reduced word error rates (WERs) with all the seeds. To further show our method has statistically significant gains, we apply Matched Pairs Sentence-Segment Word Error significance tests (mapsswe) [36] for the best Transformer models. On both SWBD/Callhome test sets, $p < 0.001$.

We further use our method to train Conformer models on SWBD. Noticing that in the Transformer experiments the probability values which lead to the best models are within the range of 0.1 to 0.2, we test use 0.1 or 0.2 as the removal probability. That is, we test not considering the total number of attention heads when setting the removal probability. In addition, we augment SWBD by speed perturbation [37] with ratio 0.9 and 1.1 and decode with a recurrent neural network language model (RNNLM) [38]. Table 4 shows our method is effective in this set of experiments. We train Conformer models using a fixed random seed. The only random factor is how the attention heads are removed in each training update and the performance of the baseline is constant. We apply a two-tailed t-test to examine if the mean WERs of the five runs of the conformer models trained with removal probability 0.2 are significantly different from the baseline Conformer. On SWBD $p < 0.003$ and on Callhome $p < 0.02$, which shows the gains are statistically significant.

Table 4: WERs on SWBD for Conformer models. * indicates the number is the average of 5 runs. In these five runs, the WERs on SWBD/Callhome ranges from 6.7 to 6.8/13.5 to 13.9.

Removal Probability	SWBD/Callhome (WER%)
Conformer Models	
0.0 (baseline)	8.4/17.1
0.1	8.2/ 16.4
0.2	8.1/16.4
+Speed Perturbation + RNNLM	
0.0 (baseline)	6.9/14.0
0.1	6.4 /14.0
0.2	6.7*/ 13.7*
Conformer [39]	7.1/15.0

Table 5: CERs on AISHELL of stochastic head removal.

Removal Probability	Dev/Test (CER%)
Transformer Models	
0.0 + speed perturbation + RNNLM	6.0/6.5
0.1 + speed perturbation + RNNLM	5.8/6.3
0.2 + speed perturbation + RNNLM	5.8/6.3
Transformer [4]	6.0/6.7
+ 5,000 hours pretrain [6]	-/6.26
Memory equipped self-attention [40]	5.74/6.46
Conformer Models	
0.0 + speed perturbation + RNNLM	5.3/6.0
0.1 + speed perturbation + RNNLM	5.2/5.8
Conformer [39]	4.4/4.7

Further, to the best of our knowledge, we have achieved the SOTA for end-to-end Transformer based models on SWBD.

Table 5 shows the results on AISHELL. We have applied mapsswe. On the Dev set, $p < 0.001$ for both the Transformer and Conformer models. On the Test set, $p = 0.030$ for the Transformer and $p = 0.023$ for the Conformer. We notice there is a performance gap between our Conformer models and the previous work [39]. This is because of different training configurations. We did not further tune our Conformer models on AISHELL due to the large number of experiments and we have indeed obtained statistically significant gains.

We use AMI to test our method on a more difficult ASR task. We use the individual headset microphone (IHM) setup. Table 6 shows our method has outperformed the baselines. Our method has achieved the SOTA for Transformer based models on AMI. We also applied mapsswe and $p < 0.001$ for both Transformer and Conformer models on both dev and test sets. Thus, we conclude on all the datasets, our method have offered statistically significant performance gains.

We noticed that the performance of our method varies among different removal probabilities. However, the probabilities between 0.1 and 0.2 have given the best accuracies. We propose to use a development set to choose the probability or using an ensemble of models trained with different probabilities as the final model, and we left these as further works.

We further study the effect of the proposed method. We have plotted the training loss of the baseline Conformer and the Conformer trained with removal probability 0.1 on SWBD. Figure 2 (a) shows the train loss curves. The proposed method has higher train loss in each epoch. This is as expected since in each training update only a fraction of the network is trained. We also compute the averaged similarity of the attention matrices of each attention head pair in each layer. We use the averaged cosine similarity of rows of the same indices as the similarity

Table 6: WERs on AMI for stochastic head removal.

Removal Probability	dev/test (WER %)
Transformer Models	
0.0 (baseline)	24.9/25.8
0.1	24.2/24.6
0.2	24.5/24.9
Conformer Models	
0.0 (baseline)	24.9/25.8
0.1	24.1/24.2
2D Conv [41]	24.9/24.6
TDNN [41]	25.3/26.0

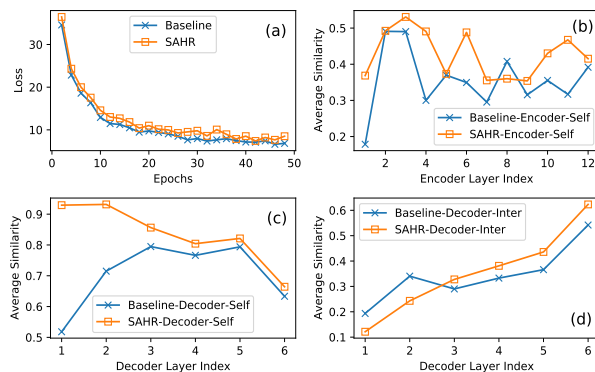


Figure 2: The train loss and the averaged similarity between attention heads of the baseline Conformer model and the Conformer model trained with stochastic attention head removal (SAHR) probability 0.1. "decoder-inter" denotes the MHA component that interacts between the encoder and the decoder.

of attention matrices since each row vector shows how the attention weights are distributed to encode the input at the row index. Figure 2 (b) show that with the random removal, the similarities between attention heads increase. Thus, it indicates the proposed method forces each attention head to extract the most essential patterns while allowing different heads to encode individual additional information. The baseline model has a smaller similarity between attention heads, further indicating each layer relies on some attention heads to learn the most useful features while other heads encode unessential information. Therefore, the proposed method better utilizes the representation powers of the multi-head attention and gives improved accuracies.

6. Conclusion

We observe there are redundant attention heads in Transformer based ASR models. Instead of searching for the best structure (which is time prohibitive), we propose to randomly remove attention heads during training and keeping all the attention heads at testing. The proposed method also forces each attention head to learn the key patterns, and thus better exploits the representation power of the multi-head structure. Our method has offered statistically significant improvements for Transformer and Conformer models on datasets of different ASR scenarios. Further work includes ensembling the models trained with different removal probabilities. Source code: https://github.com/s1603602/attention_head_removal

7. Acknowledgements

SZ was supported by a PhD Studentship funded by Toshiba Research Europe Limited.

8. 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 *NeurIPS*, 2017.
- [2] L. Dong, S. Xu, and B. Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *ICASSP*, 2018.
- [3] N.-Q. Pham, T.-S. Nguyen, J. Niehues, M. Müller, and A. Waibel, "Very deep self-attention networks for end-to-end speech recognition," *INTERSPEECH*, 2019.
- [4] 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," in *ASRU*, 2019.
- [5] K. J. Han, J. Huang, Y. Tang, X. He, and B. Zhou, "Multi-stride self-attention for speech recognition," in *INTERSPEECH*, 2019.
- [6] D. Jiang, W. Li, R. Zhang, M. Cao, N. Luo, Y. Han, W. Zou, and X. Li, "A further study of unsupervised pre-training for transformer based speech recognition," *arXiv preprint arXiv:2005.09862*, 2020.
- [7] Z. Tian, J. Yi, Y. Bai, J. Tao, S. Zhang, and Z. Wen, "Synchronous transformers for end-to-end speech recognition," in *ICASSP*, 2020.
- [8] 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," in *INTERSPEECH*, 2020.
- [9] H. Luo, S. Zhang, M. Lei, and L. Xie, "Simplified self-attention for transformer-based end-to-end speech recognition," *arXiv preprint arXiv:2005.10463*, 2020.
- [10] 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," in *INTERSPEECH*, 2020.
- [11] S. Zhang, E. Loweimi, P. Bell, and S. Renals, "On the usefulness of self-attention for automatic speech recognition with transformers," in *IEEE SLT*, 2021.
- [12] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *ACL*, 2019.
- [13] D. B. Paul and J. M. Baker, "The design for the wall street journal-based csr corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992.
- [14] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *O-COCOSDA*, 2017.
- [15] J. Godfrey, E. Holliman, and J. McDaniel, "Switchboard: telephone speech corpus for research and development," in *ICASSP*, 1992.
- [16] S. Renals, T. Hain, and H. Bourlard, "Recognition and understanding of meetings the ami and amida projects," in *ASRU*, 2007.
- [17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *JMLR*, 2014.
- [18] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one," in *NeurIPS*, 2019.
- [19] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," in *ICLR*, 2019.
- [20] H. Peng, R. Schwartz, D. Li, and N. A. Smith, "A mixture of $h-1$ heads is better than h heads," *ACL*, 2020.
- [21] W. Zhou, T. Ge, K. Xu, F. Wei, and M. Zhou, "Scheduled drophead: A regularization method for transformer models," in *EMNLP*, 2020.
- [22] Z. Wu, Z. Liu, J. Lin, Y. Lin, and S. Han, "Lite transformer with long-short range attention," in *ICLR*, 2020.
- [23] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.
- [24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997.
- [25] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *ASRU*, 2011.
- [26] P. Ghahremani, B. BabaAli, D. Povey, K. Riedhammer, J. Trmal, and S. Khudanpur, "A pitch extraction algorithm tuned for automatic speech recognition," in *ICASSP*, 2014.
- [27] 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," *INTERSPEECH*, 2019.
- [28] L. Lu, C. Liu, J. Li, and Y. Gong, "Exploring transformers for large-scale speech recognition," in *INTERSPEECH*, 2020.
- [29] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," in *ACL*, 2019.
- [30] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," *INTERSPEECH*, 2018.
- [31] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NeurIPS 2017 Workshop Autodiff*, 2017.
- [32] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.
- [33] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *ACL*, 2016.
- [34] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.
- [35] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *ICASSP*, 2017.
- [36] L. Gillick and S. Cox, "Some statistical issues in the comparison of speech recognition algorithms," in *ICASSP*, 1989.
- [37] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *INTERSPEECH*, 2015.
- [38] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint ctc-attention based end-to-end speech recognition with a deep cnn encoder and rnn-lm," in *INTERSPEECH*, 2017.
- [39] P. Guo, F. Boyer, X. Chang, T. Hayashi, Y. Higuchi, H. Inaguma, N. Kamo, C. Li, D. Garcia-Romero, J. Shi, J. Shi, S. Watanabe, K. Wei, W. Zhang, and Y. Zhang, "Recent developments on espnet toolkit boosted by conformer," *arXiv preprint arXiv:2010.13956*, 2020.
- [40] Z. Gao, S. Zhang, M. Lei, and I. McLoughlin, "San-m: Memory equipped self-attention for end-to-end speech recognition," in *INTERSPEECH*, 2020.
- [41] D. Oglic, Z. Cvetkovic, P. Bell, and S. Renals, "A deep 2d convolutional network for waveform-based speech recognition," in *INTERSPEECH*, 2020.