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

Transformer-based ASR Incorporating Time-reduction Layer and Fine-tuning with Self-Knowledge Distillation

Md Akmal Haidar, Chao Xing, Mehdi Rezagholizadeh

Huawei Noah's Ark Lab, Montreal Research Centre, Canada
{md.akmal.haidar, xingchao.ml, mehdi.rezagholizadeh}@huawei.com

Abstract

Reducing the input sequence length of speech features to alleviate the complexity of alignment between speech features and text transcript by sub-sampling approaches is an important way to get better results in end-to-end (E2E) automatic speech recognition (ASR) systems. This issue is more important in Transformer-based ASR, because the self-attention mechanism in Transformers has $O(n^2)$ order of complexity in both training and inference. In this paper, we propose a Transformer-based ASR model with the time-reduction layer, in which we incorporate time-reduction layer inside transformer encoder layers in addition to traditional sub-sampling methods to input features that further reduce the frame-rate. This can help in reducing the computational cost of the self-attention process for training and inference with performance improvement. Moreover, we introduce a fine-tuning approach for pre-trained ASR models using self-knowledge distillation (S-KD) which further improves the performance of our ASR model. Experiments on LibriSpeech datasets show that our proposed methods outperform all other Transformer-based ASR systems. Furthermore, with language model (LM) fusion, we achieve new state-of-the-art word error rate (WER) results for Transformer-based ASR models with just 30 million parameters trained without any external data.

Index Terms: speech recognition, transformer, sub-sampling, self-knowledge distillation

1. Introduction

End-to-end (E2E) automatic speech recognition (ASR) systems [1, 2, 3, 4] have shown great success recently because of their simple training and inference procedures over the traditional HMM-based methods [5]. These E2E models learn a direct mapping of the input acoustic signal to the output transcription without needing to decompose the problem into different parts such as lexicon modeling, acoustic modeling and language modeling. The very first E2E ASR model is the connectionist temporal classification (CTC) [1] model which independently maps the acoustic frames into the outputs. The conditional independence assumption in CTC was tackled by the recurrent neural network transducer (RNNT) model [6, 7], which shows better performance in streaming scenarios. Other E2E ASR models are the attention-based encoder-decoder (AED) architectures which yield state-of-the-art results in offline and online fashions [8, 4, 3, 9, 10, 11, 12]. The Transformer architecture, which uses self-attention mechanism to model temporal contextual information, has been shown to achieve lower word error rate (WER) compared to the RNN-based AED architectures [4]. However, Transformer architectures suffer from the decreased computational efficiency for longer input sequences because of quadratic time requirements of the self-attention mechanism.

For ASR systems, the number of time frames for an audio input sequence is significantly higher than the number of output text labels. Considering the adjacent speech frames can form a chunk to represent more meaningful units like phonemes, some pre-processing mechanisms are considered to capture the embedding of a group of speech frames to reduce the frame-rate in the encoder input. In [13], different pre-processing strategies such as convolutional sub-sampling and frame stacking & skipping techniques for Transformer-based ASR are discussed. Among these approaches, the convolutional approach for frame-rate reduction gives better WER results compared to other approaches. Recently, a VGG-like convolutional block [14] with max-pooling [15] and layer normalization is used [11] before the Transformer encoder layers and outperforms the 2D-convolutional sub-sampling [4]. In [3], a pyramidal encoder structure for Bidirectional LSTM (BLSTM) is proposed using time-reduction layers to reduce the frame-rate of the input sequence by concatenating adjacent frames. In [7], a time-reduction layer is employed in the RNNT model to speed up the training and inference. However, time-reduction layer with convolutional sub-sampling for Transformer architecture was never explored.

In this work, we hypothesize that the hidden representation of low frame-rate features would carry meaningful representations. In this regard, we incorporate a time-reduction layer by concatenating adjacent frames on top of convolution sub-sampling approaches for Transformer-based ASR models that further decreases the frame-rate. Our proposed approach can speed up the Transformer model training and inference. Also, our model yields new state-of-the art WER results for Transformer-based ASR models over traditional approaches. Furthermore, we introduce a fine-tuning approach for pre-trained ASR models incorporating the self-knowledge distillation (S-KD) approach [16, 17], which further improves the performance of our ASR model.

2. Related Work

Transformer ASR Several studies have focused on adapting Transformer networks for end-to-end speech recognition. In particular, [18, 19] present models augmenting Transformer networks with convolutions. [4] focuses on refining the training process, and show that Transformer-based end-to-end ASR is highly competitive with state-of-the-art methods. In [11], a regularization method based on semantic masking was introduced for Transformer ASR. A hybrid Transformer model with deeper layers and iterative loss was introduced in [13]. In [20], a semi-supervised model with pseudo-labeling using Transformer-based acoustic model was introduced. Recently a convolution augmented Transformer was proposed [21] where a convolution module with two macaron-like feed forward layers is augmented in the Transformer encoder layers.

Knowledge Distillation Knowledge distillation (KD) is a prominent neural model compression technique [22] in which the output of a teacher network is used as an auxiliary supervision besides the ground-truth training labels. Later on, it was shown that KD can be used for improving the performance of neural networks in the so-called born-again [23] or self-distillation frameworks [16, 24]. Self-distillation is a regularization technique trying to improve the performance of a network using its internal knowledge. While KD has shown great success in different ASR tasks [25, 26, 27, 28, 29, 30, 31], self-distillation is more investigated in computer vision and natural language processing (NLP) domains [17, 24]. To the best of our knowledge, we incorporate the self-KD approach for the first time in training ASR models.

3. Proposed Model for Transformer ASR

Given an input sequence X of log-mel filterbank speech features, Transformer predicts a target sequence Y of characters or SentencePiece [32]. The architecture of our proposed model is described in Figure 1.

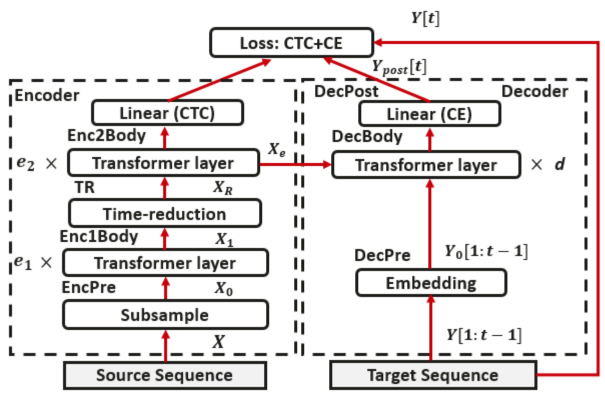

Figure 1: *Proposed Transformer ASR with time-reduction layer*

3.1. Transformer Architecture

In Figure 1, EncPre(.) transforms the source sequence X into a sub-sampled sequence $X_0 \in R^{n_{sub} \times d_{att}}$ by using convolutional neural network (CNN) blocks [4, 10, 11] which reduce the sequence length of the output X_0 by a factor of 4 compared to the sequence length of X. In the baseline model, the EncPre(.) is followed by a stack of Transformer layers that transform X_0 into a sequence of encoded features $X_e \in R^{n_{sub} \times d_{att}}$ for the CTC and decoder networks [4]. In this paper, we incorporate a time-reduction (TR) layer in the stack of Transformer layers of the encoder. The position of the TR layer in the Transformer layers is a hyper-parameter. In Figure 1, we add a TR layer between two groups (Enc1Body(.) and Enc2Body(.)) of Transformer layers in the encoder. The Enc1Body(.) transforms X_0 into a sequence of encoded features $X_1 \in R^{n_{sub} \times d_{att}}$ for the TR layer. The TR layer transforms the sequence X_1 into $X_R \in R^{n_{sub2} \times d_{att}}$ by concatenating two adjacent time frames [3]. This concatenation further reduces the length of the output sequence X_R by a factor of 2 (i.e., the frame-rate of X_R is reduced by a factor of 8 compared to X). Here, n_{sub2} represents the sequence length after applying a TR layer. A TR layer corresponding to the j^{th} layer at the output of i^{th} time step can be described as follows [3]:

$$h_i^j = \left[h_{2i}^{j-1}, h_{2i+1}^{j-1}\right] \tag{1}$$

where h_i^j represents the encoder representation of the j^{th} layer at the i^{th} time step. After the TR layer, Enc2Body(.) transforms X_R into a sequence of encoded features $X_e \in R^{n_{sub2} \times d_{att}}$ for the CTC and decoder networks. The Enc1Body(.) and Enc2Body(.) can be defined as [4]:

$$\begin{aligned} X_i^{'} &= X_i + MHA_i(X_i, X_i, X_i) \\ X_{i+1} &= FFN_i(X_i^{'}) \end{aligned} \tag{2}$$

where $i = 0, \ldots, e_1 - 1$ and $i = 0, \ldots, e_2 - 1$ describe the index of the Transformer layers of the encoder before and after the TR layer respectively.

We keep the same decoder architecture as in [4, 8]. The decoder receives the encoded sequence X_e and the prefix of a target sequence $Y[1:t-1]$ of token IDs: characters or SentencePiece [32]. First, DecPre(.) embeds the tokens into learnable vectors $Y_0[1:t-1]$. Then, DecBody(.) and a single linear layer DecPost(.) predicts the posterior distribution of the next token prediction $Y_{post}[t]$ given X_e and $Y[1:t-1]$. For the decoder input $Y[1:t-1]$, we use ground-truth labels in the training stage, while we use generated output in the decoding stage. The DecBody(.) can be described as:

$$\begin{aligned} Y_j^{'}[t] &= Y_j[t] + MHA_j^{self}(Y_j[t], Y_j[1:t-1], Y_j[1:t-1]) \\ Y_j^{''} &= Y_j + MHA_j^{src}(Y_j^{'}, X_e, X_e) \\ Y_{j+1} &= Y_j^{''} + FFN_j(Y_j^{''}) \end{aligned} \tag{3}$$

where $j = 0, \ldots, d-1$ represents the index of the Transformer layers of the decoder. $MHA_j^{src}(Y_j^{'}, X_e, X_e)$ and $MHA_j^{self}(Y_j[t], Y_j[1:t-1], Y_j[1:t-1])$ are defined as the 'encoder-decoder attention' and the 'decoder self-attention' respectively [33, 4].

The details of the MHA, FFN, and other components of the architecture such as sinusoidal positional encodings, residual connections and layer normalization can be found in [33, 4]. The positional encodings are applied into X_0 and Y_0 when 2D-convolutional sub-sampling was used [4, 10]. For VGG-like convolutional sub-sampling [11], the positional encoding for X_0 was discarded. We define 2D-convolution sub-sampling [4, 10] and VGG-like convolutional sub-sampling [11] for EncPre(.) as CONV2D4 and VGGCONV2D4 respectively. For both cases, the sequence length reduces by a factor of 4 compared to the length of X.

3.2. Training and Decoding

During ASR training, the frame-wise posterior distribution of $P_{s2s}(Y|X)$ and $P_{ctc}(Y|X)$ are predicted by the decoder and the CTC module respectively. The training loss function is the weighted sum of the negative log-likelihood of these values [4]:

$$L_{ASR} = -(1-\alpha)\log P_{s2s}(Y|X) - \alpha \log P_{ctc}(Y|X) \tag{4}$$

where $-\log P_{ctc}(Y|X)$ and $-\log P_{s2s}(Y|X)$ describe the CTC and cross-entropy (CE) losses respectively. α is a hyper-parameter which determines the CTC weight. In the decoding stage, given the speech feature X and the previous predicted token, the next token is predicted using beam search combining the scores of s2s and ctc with/without language model score following [4].

4. Proposed Fine-tuning Approach using Self-KD

We further improve the performance of our proposed model by incorporating self-knowledge distillation (S-KD) approach [16]. S-KD has recently been investigated for natural language processing (NLP) applications, where the soft labels generated by the model are used to improve itself in a progressive manner [17, 16].

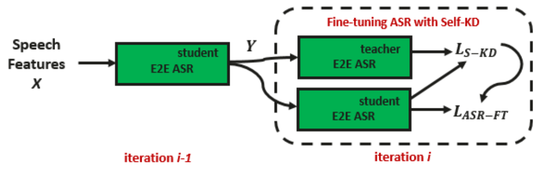

Figure 2: *Fine-tuning an E2E ASR model using the S-KD approach. The student at iteration $i-1$ becomes the teacher at iteration i and the model is trained with the loss L_{ASR-FT}*

S-KD is a framework of KD where the student becomes its own teacher. In this paper, we incorporate S-KD in fine-tuning a pre-trained ASR model and will show that it will be more efficient than training ASR models using S-KD from scratch. For fine-tuning using S-KD, let $P^T_{s2s}(Y[t]|Y[1:t-1], X_e)$ be the probability distribution of the decoder of a pre-trained ASR model as a teacher. We initialized the student with the teacher model and at iteration $i-1$, the student model tries to mimic the teacher probability distribution with cross-entropy:

$$L_{S-KD} = -\sum_{Y[t]\in V} (P^T_{s2s}(Y[t]|Y[1:t-1], X_e)\times \log P_{s2s}(Y[t]|Y[1:t-1], X_e)) \quad (5)$$

where V is a set of vocabulary. At iteration i, the student probability distribution $P_{s2s}(Y[t]|Y[1:t-1], X_e)$ of the decoder becomes the teacher probability distribution $P^T_{s2s}(Y[t]|Y[1:t-1], X_e)$ and the student model is trained to mimic the teacher with L_{S-KD}. A schematic of the S-KD approach is described in Figure 2. The new loss function for fine-tuning (FT) the ASR model can be described as:

$$L_{ASR-FT} = \alpha L_{ctc} + (1-\alpha)(\phi_{KD} L_{S-KD} + (1-\phi_{KD}) L_{s2s}) \quad (6)$$

where $L_{ctc} = -\log P_{ctc}(Y|X)$, $L_{s2s} = -\log P_{s2s}(Y|X)$, and ϕ_{KD} is a controlled parameter, and is typically $\phi_{KD} = 0.5$ [34]. To show the effectiveness of our proposed fine-tuning approach using S-KD, we also perform S-KD experiments to train our model from scratch. In this regard, we update the KD loss for each epoch similar to [16]:

$$\phi_{KD_t} = \phi_{KD_T} \times \frac{t}{T} \quad (7)$$

where T is the total number of training epoch~~s~~, ϕ_{KD_T} is the ϕ_{KD_t} for the last epoch.

5. Experiments

5.1. Experimental Setup

We use the open-source, ESPnet toolkit [35] for our experiments. We conduct our experiments on LibriSpeech dataset [36], which is a speech corpus of reading English audio books. It has 960 hours of training data, 10.7 hours of development data, and 10.5 hours of test data, whereby the development and the test datasets are both split into approximately two halves named "clean" and "other". To extract the input features for speech, we follow the same setup as in [35, 4]: using 83-dimensional log-Mel filterbanks frames with pitch features [4, 10]. The output tokens come from a 5K subword vocabulary created with sentencepiece [32]"unigram" [35]. We perform experiments using 12 ($e_1+e_2 = 12$) and 6 ($d = 6$) Transformer layers for the encoder and decoder respectively, with $d_{ff} = 2048$, $d_{att} = 256$, and $h = 4$. The total number of model parameters are about 30 M. We apply default settings of SpecAugmentation [37] of ESPnet [35] and no gradient accumulation for all of our experiments.

We use Adam optimizer with learning rate scheduling similar to [33, 35] and other settings (e.g., dropout, warmup steps, α, label smoothing penalty [37]) following [10, 35]. We set the initial learning rate of 5.0 [35]. We train all of our models for 150 epochs. We report our results based on averaging the best five checkpoints. For fine-tuning with self-knowledge distillation (FS-KD) experiments, we use our trained best average model for initialization. We train the FS-KD experiments for 50 epochs with Adam optimizer [38] with fixed learning rate of 0.0001 [35] and fixed $\phi_{KD} = 0.5$. We also perform S-KD experiments from scratch for 150 epochs with $\phi_{KD_T} = 0.5$ and 200 epochs with $\phi_{KD_T} = 0.7$ respectively. For S-KD experiments, We use Adam optimizer with learning rate scheduling similar to [33, 35].

For decoding, the CTC weight λ, the LM weight γ and beam size are 0.5, 0.7 and 20 respectively [10, 35]. Moreover, we apply insertion-penalty of 2.0 during decoding. For S-KD and FS-KD experiments, we note that the CTC weight of $\lambda = 0.3$ gives better results. This might be because of S-KD applied on the decoder output of the ASR model. For LM fusion [39], we use a pre-trained Transformer LM[1].

Table 1: *WER results for E2E ASR models. TR0 and TR2 represent the time-reduction (TR) layer applied before all and after second Transformer layers of the encoder respectively. Best results are depicted in bold.*

Model	Test-clean	Test-other
Hybrid Transformer+LM [13]	2.26	4.85
RWTH+LM [40]	2.3	5.0
Baseline +LM [10]	2.7	6.1
ESPNet Transformer+LM [4]	2.6	5.7
LAS+LM [37]	3.2	9.8
LAS+SpecAugment+LM [37]	2.5	5.8
Transformer+LM [20]	2.33	5.17
Semantic Mask + LM [11]	2.24	5.12
Transformer Transducer+LM [41]	2.0	4.6
New Baselines		
Pyramidal Transformer	4.3	9.3
+ LM	2.9	5.9
CONV2D8	4.1	9.1
+ LM	2.7	5.6
VGGCONV2D8	3.6	9.2
+ LM	2.0	5.3
Proposed Models with TR		
CONV2D4+TR0	3.6	8.6
+ LM	2.3	5.3
VGGCONV2D4+TR0	3.5	8.5
+ LM	**2.0**	**5.0**
CONV2D4+TR2	3.7	8.2
+ LM	2.4	5.2
VGGCONV2D4+TR2	3.3	8.5
+ LM	**2.0**	**5.0**

5.2. New Baseline Results

To compare with our proposed approach, we use CONV2D8 sub-sampling [35] and VGGCONV2D8 [11, 35] for EncPre(.) to reduce the sequence length by a factor of 8 compared to the length of X and then apply all the Transformer layers. Furthermore, we create a pyramidal structure [3] in the first three Transformer layers of the encoder, where we concatenate the outputs at consecutive steps (Equation 1) of each layer before

[1]https://github.com/espnet/espnet/blob/master/egs/librispeech/asr1/RESULTS.md

feeding it to the next layer. After the first three layers, the sequence length would reduce by a factor of 8 than the sequence length of X. We define the model as Pyramidal Transformer. From Table 1, we can see that our new baseline models give comparable results to the existing models with LM fusion, which are much larger than our model. Among these baselines, the VGGCONV2D8 gives the better results.

5.3. Results using TR layer

We apply the TR layer after CONV2D4 or VGGCONV2D4 to further reduce the sequence length by a factor of two. First, we apply the TR layer after CONV2D4 and VGGCONV2D4 approaches (i.e., TR0: $e_1 = 0$ and $e_2 = 12$). We define the models as CONV2D4+TR0 and VGGCONV2D4+TR0 respectively, which reduce the frame-rate by a factor of 8 than the frame-rate of X. Next, we apply the TR layer after the second Transformer layer (i.e., TR2: $e_1 = 2$ and $e_2 = 10$). From both TR0 and TR2 experiments, we can see that both experiments give comparable results and VGGCONV2D4+TR0/TR2 yields better results over the CONV2D4+TR0/TR2 models. Without LM fusion, VGGCONV2D4+TR2 gives better results over VGGCONV2D4+TR0. For the other experiments, we use the VGGCONV2D4+TR2 setting. From Table 1, we can note that VGGCONV2D4+TR0/TR2 model with LM fusion outperforms or gives comparable results to the existing hybrid results [12, 40], E2E LSTM [37] and E2E Transformer baselines [10, 4, 20, 11] which are trained with more parameters. [10, 4] and [11] use CONV2D4 and VGGCONV2D4 respectively for EncPre(.). Compared to the best E2E transformer ASR using semantic mask technique [11] and [20], our proposed approach achieves relative WER reductions of 10.7% and 14.1% for the test-clean and 2.3% and 3.2% for the test-other respectively. Also, our approach shows comparable results to Transformer Transducer [41] which is trained with RNN-T loss [42] and 139 M parameters.

Table 2: *WER results of the proposed Transformer-based ASR models using self-KD. Best results are described in bold.*

Model	Test clean	Test other
VGGCONV2d4+TR2 +FS-KD	3.1	7.9
+ LM	**1.9**	**4.8**
VGGCONV2d4+TR2_S-KD*	3.2	8.1
+ LM	2.0	4.8
VGGCONV2d4+TR2_S-KD*+FS-KD	3.3	8.1
+ LM	2.0	4.8
VGGCONV2d4+TR2_S-KD**	3.2	8.0
+ LM	2.0	4.9

5.4. Results using Self-KD

We report the results for self-KD (S-KD) experiments in Table 2. We conduct experiments VGGCONV2D4+TR2+FS-KD for fine-tuning the pre-trained ASR model VGGCONV2D4+TR2 using S-KD. We can note that with FS-KD, the proposed method gives further WER reductions for both without and with LM fusion. With LM fusion, our model outperforms the Transformer transducer model for the test-clean dataset and gives comparable results for the test-other dataset. We also train an ASR model by applying S-KD from scratch to show the effectiveness of our fine-tuning approach FS-KD. We conduct two experiments by applying self-KD from scratch for 150 epochs (VGGCONV2D4+TR2_S-KD*) and 200 epochs (VGGCONV2D4+TR2_S-KD**) respectively. The S-KD experiments from scratch give similar or better results than the VGGCONV2D4+TR2 model but lower performance than our proposed VGGCONV2D4+TR2+FS-KD approach. Also, the self-KD approach from scratch requires longer training time than our proposed approaches. Furthermore, to make the same number of epochs, we apply self-KD for fine-tuning the VGGCONV2D4+TR2_S-KD* model which is trained using the self-KD from scratch for 150 epochs. It can be seen that the model VGGCONV2D4+TR2_S-KD*+FS-KD does not give better results than our proposed model VGGCONV2D4+TR2+FS-KD. Moreover, we report the best five validation accuracy for our best model with the time-reduction layer and the models with self-KD in Table 3. We observe that our proposed fine-tuning approach using self-KD (VGGCONV2D4+TR2+FS-KD) gives better validation accuracy over the other models.

Table 3: *Rows 1-5 represent the best five validation accuracy for the VGGCONV2D4+TR2, VGGCONV2D4+TR2+FS-KD, VGGCONV2D4+TR2+S-KD*, VGGCONV2D4+TR2_S-KD*+FS-KD, and VGGCONV2D4+TR2_S-KD**. Best results are in bold.*

1	2	3	4	5
0.95053	0.94900	0.94793	0.94780	0.94735
0.95518	**0.95475**	**0.95436**	**0.95434**	**0.95418**
0.95043	0.95011	0.95002	0.94999	0.94966
0.95209	0.95184	0.95072	0.95049	0.94973
0.95266	0.95216	0.95189	0.95187	0.95146

5.5. Time Complexity Analysis

Transformers are recently the most efficient models which are suffered from a $(1 + \frac{m}{l})^2$ decay in computational efficiency for the self-attention mechanism if the sequence length l is increased by m steps. Our proposed approach could release the burden of computational efficiency in Transformer ASR by applying time-reduction layer with convolutional sub-sampling approaches and gives significant improvement over traditional approaches. The proposed method can reduce the computational cost of the self-attention mechanism by k^2 times over the traditional approach, where k is the time-reduction ratio.

6. Conclusion and Future Work

Convolutional sub-sampling approaches are essential for efficient use of self-attention mechanism in Transformer-based ASR. These sub-sampling approaches reduce the speech frame-rate to form meaningful units like phoneme, word-piece, etc. The reduced frame-rate can help in proper training of Transformer ASR. In this paper, we incorporated a time-reduction layer for Transformer-based ASR which can further reduce the frame-rate and improve the performance over traditional convolutional sub-sampling approaches. It can also gives faster training and inference during the self-attention computation of the Transformer ASR model. Our approach yields new state-of-the-art results using the LibriSpeech benchmark dataset for Transformer architectures. Furthermore, we introduced fine-tuning of a pre-trained ASR model using self-knowledge distillation which further improves the performance of our ASR model. Moreover, we showed our proposed fine-tuning approach with self-KD outperformed the conventional self-KD training from scratch. We performed experiments on LibriSpeech datasets and show that our proposed method with 30 M parameters outperformed over the other Transformer-based ASR models. For future work, we will investigate our approaches for conformer [21] architecture and explore more experimental settings with incorporating gradient accumulation.

7. References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006, pp. 369–376.

[2] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *ICML*, 2016, pp. 173–182.

[3] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *ICASSP*, 2016, pp. 4960–4964.

[4] S. Karita, N. Chen, T. Hayashi, T. Hori, K. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, "A comparative study on transformer vs rnn in speech applications," in *IEEE 2019 ASRU Workshop*, 2019.

[5] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi," in *INTERSPEECH*, 2016.

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

[7] Y. He, T. N. Sainath, R. Prabhavalkar, I. Mcgraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, and R. Pang et al., "Streaming end-to-end speech recognition for mobile devices." in *ICASSP*, 2019.

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

[9] M. A. Haidar and M. Rezagholizadeh, "Fine-tuning of pre-trained end-to-end speech recognition with generative adversarial networks," in *ICASSP*, 2021.

[10] N. Moritz, T. Hori, and J. L. Roux, "Streaming automatic speech recognition with the transformer model," in *ICASSP*, 2020.

[11] C. Wang, Y. Wu, Y. Du, J. Li, S. Liu, L. Lu, G. Ren, S. Ye, S. Zhao, and M. Zhou, "Semantic mask for transformer based end-to-end speech recognition," *arXiv preprint arXiv:1912.03010*, 2020.

[12] C. Wang, Y. Wu, L. Lu, S. Liu, J. Li, G. Ye, and M. Zhou, "Low latency end-to-end streaming speech recognition with a scout network," *arXiv preprint arXiv:2003.10369*, 2020.

[13] 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," in *ICASSP*, 2020.

[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[15] 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.

[16] K. Kim, B. M. Ji, D. Yoon, and S. Hwang, "Self-knowledge distillation: A simple way for better generalization," *arXiv preprint arXiv:2006.12000v1*, 2020.

[17] S. Haun and H. Choi, "Self-knowledge distillation in natural language processing," in *Recent Advances in Natural Language Processing (RANLP)*, 2019.

[18] L. Dong, S. Xu, and B. Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *ICASSP*, 2018, pp. 5884–5888.

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

[20] G. Synnaeve, Q. Xu, J. Kahn, T. Likhomanenko, E. Grave, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, "End-to-end asr: From supervised to semi-supervised learning with modern architectures," *arXiv preprint arXiv:1911.08460*, 2020.

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

[22] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[23] T. Furlanello, Z. Lipton, M. Tschannen, L. Itti, and A. Anandkumar, "Born again neural networks," in *ICML*. PMLR, 2018, pp. 1607–1616.

[24] S. Hahn and H. Choi, "Self-knowledge distillation in natural language processing," *arXiv preprint arXiv:1908.01851*, 2019.

[25] R. Pang, T. N. Sainath, R. Prabhavalkar, S. Gupta, Y. Wu, S. Zhang, and C.-C. Chiu, "Compression of end-toend models," in *INTERSPEECH*, 2018.

[26] M. Huang, Y. You, Z. Chen, Y. Qian, and K. Yu, "Knowledge distillation for sequence model," in *INTERSPEECH*, 2018.

[27] R. Takashima, S. Li, and H. Kawai, "An investigation of a knowledge distillation method for ctc acoustic models," in *ICASSP*, 2018.

[28] H.-G. Kim, H. Na, H. Lee, J. Lee, T. Kang, M.-J. Lee, and Y. S. Choi, "Knowledge distillation using output errors for self-attention end-to-end models," in *ICASSP*, 2019.

[29] Y. Chebotar and A. Waters, "Distilling knowledge from ensembles of neural networks for speech recognition," in *INTERSPEECH*, 2016.

[30] T. Fukuda, M. Suzuki, G. Kurata, S. Thomas, J. Cui, and B. Ramabhadran, "Efficient knowledge distillation froman ensemble of teachers," in *INTERSPEECH*, 2017.

[31] J. W. Yoon, H. Lee, H. Y. Kim, W. I. Cho, and N. S. Kim, "Tutornet: Towards flexible knowledge distillation forend-to-end speech recognition," *arXiv preprint arXiv:2008.00671*, 2020.

[32] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:1808.06226*, 2018.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[34] E. Tsunoo, Y. Kashiwagi, and S. Watanabe, "Streaming transformer asr with blockwise synchronous inference," *arXiv preprint arXiv:2006.14941*, 2020.

[35] 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," *arXiv preprint arXiv:1804.00015*, 2018.

[36] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *ICASSP*, 2015, pp. 5206–5210.

[37] 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," *arXiv preprint arXiv:1904.08779*, 2019.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[39] S. Toshniwal, A. Kannan, C.-C. Chiu, Y. Wu, T. N. Sainath, and K. Livescu, "A comparison of techniques for language model integration in encoder-decoder speech recognition," in *IEEE 2018 SLT Workshop*, 2018.

[40] C. Luscher, E. Beck, K. Irie, M. Kitza, W. Michel, A. Zeyer, R. Schluter, and H. Ney, "Rwth asr systems for librispeech: Hybrid vs attention - w/o data augmentation," *arXiv preprint arXiv:1905.03072v3*, 2019.

[41] 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," *arXiv preprint arXiv:2002.02562*, 2020.

[42] A. Tripathi, H. Lu, H. Sak, and H. Soltau, "Monotonic recurrent neural network transducer and de-coding strategies," in *IEEE 2019 ASRU Workshop*, 2019.