SpecSwap: A Simple Data Augmentation Method for End-to-End Speech Recognition

Xingchen Song[1,†], *Zhiyong Wu*[1,2,‡], *Yiheng Huang*[3], *Dan Su*[3], *Helen Meng*[1,2]

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[2]The Chinese University of Hong Kong, Shatin, N.T., Hong Kong SAR, China
[3]Tencent AI Lab, Shenzhen, China

sxc19@mails.tsinghua.edu.cn, {zywu,hmmeng}@se.cuhk.edu.hk,
{arnoldhuang,dansu}@tencent.com

Abstract

Recently, End-to-End (E2E) models have achieved state-of-the-art performance for automatic speech recognition (ASR). Within these large and deep models, overfitting remains an important problem that heavily influences the model performance. One solution to deal with the overfitting problem is to increase the quantity and variety of the training data with the help of data augmentation. In this paper, we present SpecSwap, a simple data augmentation scheme for automatic speech recognition that acts directly on the spectrogram of input utterances. The augmentation policy consists of swapping blocks of frequency channels and swapping blocks of time steps. We apply SpecSwap on Transformer-based networks for end-to-end speech recognition task. Our experiments on Aishell-1 show state-of-the-art performance for E2E models that are trained solely on the speech training data. Further, by increasing the depth of model, the Transformers trained with augmentations can outperform certain hybrid systems, even without the aid of a language model.

Index Terms: end-to-end speech recognition, data augmentation

1. Introduction

End-to-end approaches have been applied successfully to Automatic Speech Recognition (ASR), where many works focus on designing deeper and larger network architectures, for example, Deep CNN [1], Deep RNN [2] and Deep Attention [3]. However, the authors in [4] have recently pointed out that overfitting is the most crucial issue when training those models on popular benchmarks.

To improve the robustness of the models and avoid overfitting, data augmentation has been proposed for ASR to produce additional training data by increasing the quantity and variety of the existing data [5, 6, 7]. Augmented data usually comes in two forms: unsupervised data and deformed data. For example, in [8], the unsupervised data was adopted by recognising it with an existing ASR system and re-training the system on both supervised and unsupervised training data. As for deformed data, simulated far-field speech [9] and clean-noisy superimposed speech [10] have been explored to supplement close-talk speech recognition. Besides, the perturbation of vocal tract length [11] and raw audio speed [12] have also been used to help ASR models to be more robust to speaker variations. More recently, borrowing the idea from computer vision [13, 14], SpecAugment [4] proposed a mask-based argumentation method that simply operates on the log-mel spectrogram of the input audio, rather than the raw audio itself. By masking blocks of consecutive time steps or mel frequency channels with zero values, they have achieved a large gain in performance of end-to-end networks.

Since SpecAugment [4] treats the log-mel spectrogram as an image and directly performs masking on it, no additional data is needed and it can be applied online during training. Inspired by this view, in this paper, we propose SpecSwap, a novel augmentation strategy which also deforms data at spectrogram-level. SpecSwap consists of two kinds of deformations of the log-mel spectrogram, namely time swapping and frequency swapping. The key idea, swapping blocks of consecutive feature vectors, comes from our previous work. In [15], we proposed an unsupervised acoustic model pretraining scheme by reconstructing frames from a permuted speech feature sequence. The permutation strategy in [15] has been proved to help the self-attention networks (SANs) generalize better. However, such permutation is achieved by modifying the attention structure and constructing special attention mask, thus it is highly bound to SANs and is difficult to adapt to other models. In addition, the SANs in [15] need to be fine-tuned after pre-training, which is more time-consuming than that of one-pass training. Given the above shortcomings, one natural question that arises is whether we can untie the model structure and permutation strategy so that the improvement of the generalization properties brought by the permutation operation can be transferred to other models at zero cost. In this paper, we address this question by applying permutation directly on spectrogram, *i.e.,* swapping blocks of features either in time-domain or frequency-domain.

While this approach is simple, it is remarkably effective and allows us to train end-to-end ASR networks, called Transformer [16], to surpass more complicated hybrid systems even without the use of Language Models (LMs). On Aishell-1, we obtain 7.60% Character Error Rate (CER) on the test set, achieving competitive result when training an end-to-end framework solely on the speech training data.

2. SpecSwap Policy

We aim to construct an augmentation policy that directly acts on the input sequences, which improves the generalization of the log-mel spectrogram encoder. A SpecSwap policy is obtained by composing two basic augmentations—frequency swapping and time swapping. Both deformations are computationally cheap and can be applied online and optimized simultaneously.

We denote the time and frequency dimensions of the spec-

[†] Work performed during an internship at Tencent AI Lab.

[‡] Corresponding author.

trogram as μ and τ respectively. A policy is made up by the following choices:

1. Frequency swapping with parameter F: A block size f is chosen from a uniform distribution from 0 to F. The consecutive log-mel frequency channels $[f_0, f_0+f)$ and $[f_1, f_1+f)$are then swapped, where f_0 and f_1 are chosen from $[0, \tau - 2f)$ and $[f_0 + f, \tau - f)$, respectively.
2. Time swapping with parameter T: A block size t is chosen from a uniform distribution from 0 to T. The consecutive time steps $[t_0, t_0 + t)$ and $[t_1, t_1 + t)$ are then swapped, where t_0 and t_1 are chosen from $[0, \mu - 2t)$ and $[t_0 + t, \mu - t)$, respectively.

Figure 1 shows examples of the individual augmentations as well as combined augmentation applied to a single input.

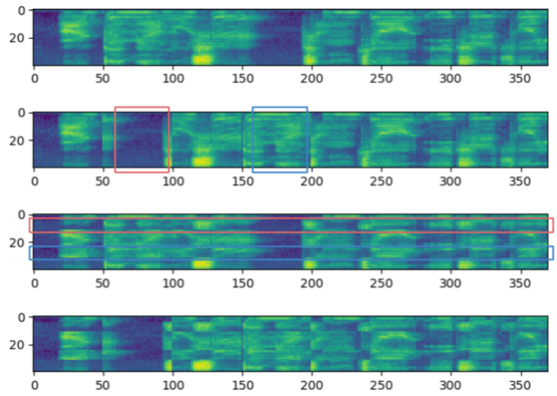

Figure 1: *Augmentations applied to the base input, given at the top. From top to bottom, the figures depict the log-mel spectrogram of the base input with no augmentation, time swapping, frequency swapping and all augmentations applied. The red box and blue box indicate the areas that were swapped in the original input.*

3. Model

In this section, we review Transformer-based networks and introduce some notations to parameterize them. We also introduce the learning rate schedules we used to train the networks because they are actually an important factor in determining performance, even more so when augmentation is applied.

3.1. Transformer Network Architectures

As shown in Figure 2, the main components of the Transformer networks [16] include an encoder, transforming the source sequence to generate a high-level representation, and a decoder generating the target sequence. The decoder models discrete tokens as a conditional language model by consuming the previously emitted characters as well as the encoder representation to calculate the probability of target sequence. Both encoder and decoder use self-attention to learn the relationship between the input and output sequence.

Instead of using a CNN for down-sampling the input spectrogram, we stack three consecutive feature vectors with a frame-skipping rate of three after applying the augmentation methods. The features are then passed through an encoder consisting of E stacked transformer blocks to yield a series of attention vectors. The attention vectors are fed into a D-layer transformer decoder, which yields the tokens for the transcript. We use same settings for both encoder blocks and decoder blocks with a per-block configuration of 4 attention heads, the model dimension $d_{model} = 256$ and feed-forward inner-layer dimension $d_{inner} = 2048$.

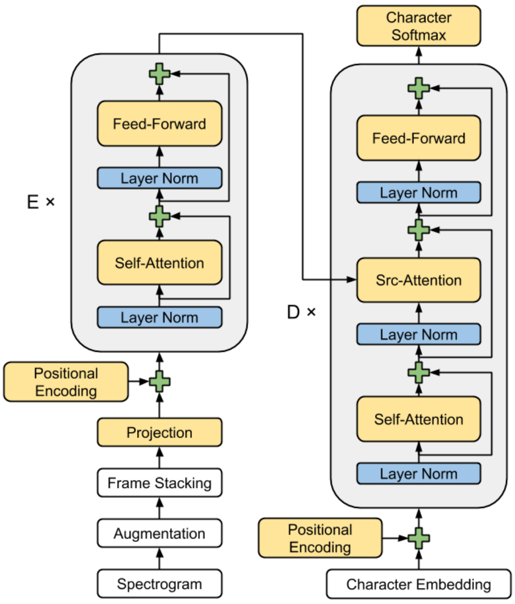

Figure 2: *Transformer architecture.*

3.2. Learning Rate Schedules

The learning rate schedule turns out to be an important factor in determining the performance of self-attention networks, especially so when augmentation is present. Here, we first introduce the training schedule that was originally described in Transformer [16]:

$$lrate = k * d_{model}^{0.5} * min(n^{-0.5}, n * warmup_n^{-1.5}) \quad (1)$$

where n is the step number, k is a fixed scalar, and the learning rate increases linearly for the first $warmup_n$ training steps and decreases thereafter proportionally to the inverse square root of the step number. The problem with this schedule is that the learning rate cannot be reduced to a suitable value within a tolerable time[1], resulting in the model still using a higher learning rate to update the parameters in the later stages of training.

To address this problem, we adopt a ***dynamic schedule***, in which the magnitude of the decrease in learning rate can be dynamically adjusted according to the performance of the model on the validation set. Formally, this schedule can be parameterized by three factors (m, v, s)-the model will perform up to m epochs and the scalar k in Eq.1 will be divided by s if validation loss doesn't decrease for v epochs during the training process.

The two dynamic schedules we use are given as the following:

1. DQ (Dynamic Quick): $(m, v, s) = (200, 4, 10)$
2. DB (Dynamic Basic): $(m, v, s) = (200, 8, 2)$

[1] Consider $k = 2.3$, $warmup_n = 20000$, at least $20G$ steps need to be conducted to reduce the learning rate from peak value $(1e-3)$ to appropriate value $(1e-6)$.

4. Experiments and Analyses

In this section, we describe our experiments on a public Mandarin speech corpus Aishell-1 [17]. The training set contains about 150 hours of speech (120,098 utterances) recorded by 340 speakers. The development set contains about 20 hours (14,326 utterances) recorded by 40 speakers. And about 10 hours (7,176 utterances) of speech is used as test set. The output alphabet of target text consists of 4233 classes, including 4230 chinese characters and three special tokens, such as <SOS>, <EOS> and <unk>.

All the acoustic features used in this paper are 40 log-mel filter-bank features extracted using Kaldi [18] with global cepstral mean and variance normalization. Beside the filter-bank features, we did not employ any auxiliary features. During training, Adam [19] optimizer ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1e-9$) is used to train our model with a batch size of 10000 frames. We choose $warmup_n = 140000, k = 11, dropout = 0.2, labelsmooth = 0.2$ for all experiments. In order to do evaluation, we save the model parameters of 5 best epochs according to the validation sets and average them at the end of training. We use character error rate (CER) as the evaluation metric. The beam search is carried out with a beam size of 15 and length penalty of 0.6. Language model was not used during decoding.

4.1. Hyperparameter Searching

To determine the best hyperparameters for spectrogram augmentation that satisfies the aforementioned definition, we conducted the following experiments. We use schedule DQ to train our baseline transformer ($E = 12, D = 6$), which is denoted as 12Enc-6Dec, on various settings of swap-related parameters. Figure 3 shows the CER of different maximum length of frequency-block, $F = 4$ and $F = 7$, at different maximum length T of time-block. We observe that SpecSwap improves the baseline result (9.98%) in all cases and the best setting ($F = 7, T = 40$) is selected for the rest of the experiments.

4.2. SpecSwap and LR Schedule

The experimental results of varying network structures, schedules and policies are shown in Table 1. A shallow configuration (*i.e.* 4-layer encoder with 2-layer decoder) is not sufficient for the task. Without using SpecSwap, the CER reduces from 15.36% to 9.11% on the test set as we increase the model depth from 04Enc-02Dec to 48Enc-24Dec. The improvement is less significant between 24Enc-12Dec and 12Enc-06Dec (only 3% relative reduction), which seems to be a symptom of overfitting. However, when using SpecSwap, we see that augmentation consistently improves the performance of the trained network, and that the benefit of schedule DB is more apparent with augmentation.

It is worth noting that for all experiments, we use not only dropout [20] but also label-smoothing [21] to prevent overfitting. The results in Table 1 reveal that SpecSwap is additive with other regularization techniques (dropout [20], label-smoothing [21]), which further improved our result to 7.60% with the 48Enc-24Dec setup.

4.3. Ablation study: SpecSwap Decomposed

So far, we have seen encouraging results. In this section, we perform an ablation study to explore the effects of each operation in SpecSwap. We ran model 12Enc-06Dec using training schedule DB for all settings (Table 2). It turns out that both swapping operations contribute to performance gain. However, frequency swapping is less effective compared with time swappping, which indicates that the major improvement comes from time-domain.

Table 1: *Aishell-1 test CER (%) evaluated for varying network structures, schedules (Sch) and policies (Pol). SS denotes SpecSwap with configuration of $F = 7$ and $T = 40$.*

Models	Size	Sch	Pol	CER (%)
04Enc-02Dec	10.61M	DQ	None	15.36
08Enc-04Dec	19.02M	DQ	None	10.83
12Enc-06Dec	27.43M	DQ	None	9.98
		DQ	SS	8.88
		DB	None	9.88
		DB	SS	8.59
24Enc-12Dec	52.66M	DQ	None	9.68
		DQ	SS	8.39
		DB	None	9.61
		DB	SS	7.85
36Enc-12Dec	68.43M	DQ	None	9.24
		DQ	SS	8.27
		DB	None	9.25
		DB	SS	7.71
48Enc-24Dec	103.12M	DQ	None	9.11
		DQ	SS	8.01
		DB	None	9.08
		DB	SS	7.60

Table 2: *Ablation study of SpecSwap, comparing frequency swapping (hyperparameter F) and time swapping (hyperparameter T).*

F	T	CER (%)
0	0	9.88
0	40	8.91
7	0	9.61
7	40	8.59

4.4. Composition Study: Integrating Other Methods

The natural next step is to determine whether we can achieve further improvements when combining other augmentation methods within a single training, *i.e.,* using audio-level augmentation method, namely Speed Perturb [12], to change the speed of the audio signals first, and then sequentially applying SpecSwap and SpecMask[2] for those spectrograms calculated from perturbed audios.

The results in Table 3 indicate that all methods help the model to generalize better and can supplement each other. This also reveals that none of them can completely solve the overfitting problem when using them alone. The End-to-End model

[2] Here we denote SpecAugment [4] as SpecMask since we only adopt time and frequency masking and discard time wrapping as suggested in [4].

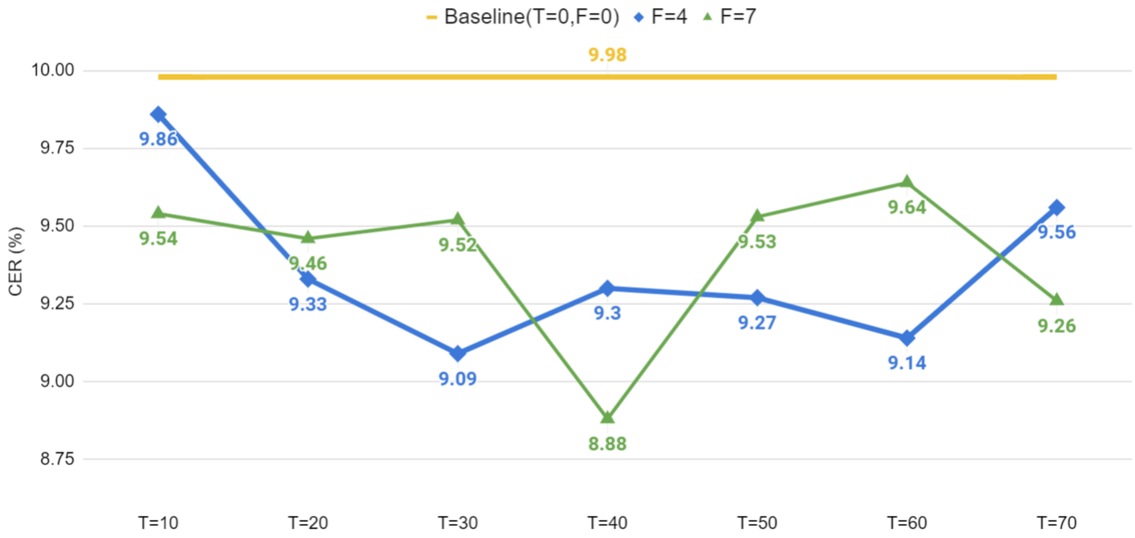

Figure 3: *CER results of hyperparameter searching.*

needs a more powerful method to fully exploit its potential and we will leave this to our future work.

Table 3: *Results on different augmentation methods. Model 12Enc-6Dec and schedule DB are used for all settings.*

Augmentation	CER (%)
None	9.88
SpecMask^2 ($F = 7, T = 40$)	8.47
SpecSwap ($F = 7, T = 40$)	8.59
Speed Perturb (speed = 0.9, 1.0, 1.1)	8.67
Composition	7.87

4.5. Comparison with Hybrid/E2E Models

Lastly, we give a CER comparison with other approaches in Table 4. For Hybrid approach, results marked with (*) were kaldi official results retrieved from the version “c7876a33”, where speaker adaptation with i-vectors were used to help the model generalize better to unseen testing speakers. For the end-to-end approach, we limit the evaluation to systems without any external data and language model fusion to examine the effectiveness of the approach. Besides, we also approximate the number of parameters based on the description in the previous studies.

In Table 4, our 24Enc-12Dec setup (with DB schedule) get 9.61% CER, which performs much better than LAS model [23] and Transformer model [24], meaning that our baseline is competitive. With SpecSwap, we further reduce CER to 7.85%, yielding a 18.3% relative reduction.

When comparing to the best hybrid models, our largest model outperforms certain systerms even without using language model. Given the potential of the models, it is strongly suggested that better results can be obtained by shallow fusion [25] and joint decoding [25] with external language model.

Table 4: *CER comparison with Hybrid and End-to-End models.*

Model	Augment	LM	Size	CER (%)
		Hybrid Approach		
kaldi/nnet3 [18] (*)	speed perturb	n-gram	-	8.64
TDNN-HMM [17]	speed perturb	n-gram	-	8.42
TDNN-LFMMI [17]	speed perturb	n-gram	-	7.62
kaldi/chain [18] (*)	speed perturb	n-gram	-	7.45
		End-to-End Approach		
RNN-T [22]	-	-	-	11.82
LAS [23]	-	-	$\approx$ 156.1M	10.56
Transformer [24]	-	-	$\approx$ 48.5M	10.00
SA-T [22]	-	-	-	9.30
		Ours		
24Enc-12Dec	-	-	52.66M	9.61
24Enc-12Dec	SpecSwap	-	52.66M	7.85
48Enc-24Dec	SpecSwap	-	103.12M	7.60

5. Conclusions and Discussions

SpecSwap greatly improves the performance of ASR networks. By utilizing swapping techniques, we are able to obtain competitive results on the Aishell-1 test set among End-to-End counterparts, surpassing the performance of certain hybrid systems even without the aid of a language model. Future work will evaluate the effectiveness of SpecSwap on industry-level large vocabulary continuous speech recognition (LVCSR) with tens of thousands hours of data. Moreover, it is also quite appealing to explore other augmentation methods to further boost the performance of End-to-End models.

6. Acknowledgements

This work was conducted when the first author was an intern at Tencent AI Lab, and is supported by joint research fund of National Natural Science Foundation of China - Research Grant Council of Hong Kong (NSFC-RGC) (61531166002, N_CUHK404/15), National Natural Science Foundation of China (61433018, 61375027). We would also like to thank Tencent AI Lab Rhino-Bird Focused Research Program (No. JR201942, JR201803) and Tsinghua University - Tencent Joint Laboratory for the support.

7. References

[1] M. Bi, Y. Qian, and K. Yu, "Very deep convolutional neural networks for LVCSR," in *Interspeech*, 2015, pp. 3259–3263.

[2] A. Graves, A. Mohamed, and G. E. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*, 2013, pp. 6645–6649.

[3] N. Pham, T. Nguyen, J. Niehues, M. Müller, and A. Waibel, "Very deep self-attention networks for end-to-end speech recognition," in *Interspeech*, 2019, pp. 66–70.

[4] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," in *Interspeech*, 2019, pp. 2613–2617.

[5] T. Nguyen, S. Stüker, J. Niehues, and A. Waibel, "Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation," in *ICASSP*, 2020, pp. 7689–7693.

[6] S. Sun, P. Guo, L. Xie, and M. Hwang, "Adversarial regularization for attention based end-to-end robust speech recognition," *IEEE/ACM TASLP*, vol. 27, no. 11, pp. 1826–1838, 2019.

[7] S. Sun, C.-F. Yeh, M. Ostendorf, M.-Y. Hwang, and L. Xie, "Training augmentation with adversarial examples for robust speech recognition," in *Interspeech*, 2018, pp. 2404–2408.

[8] A. Ragni, K. M. Knill, S. P. Rath, and M. J. F. Gales, "Data augmentation for low resource languages," in *Interspeech*, 2014, pp. 810–814.

[9] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *ICASSP*, 2017, pp. 5220–5224.

[10] A. Y. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," *CoRR*, vol. abs/1412.5567, 2014. [Online]. Available: http://arxiv.org/abs/1412.5567

[11] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *ICML*, vol. 117, 2013.

[12] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Interspeech*, 2015, pp. 3586–3589.

[13] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *CVPR*, 2019, pp. 113–123.

[14] T. Devries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *CoRR*, vol. abs/1708.04552, 2017. [Online]. Available: http://arxiv.org/abs/1708.04552

[15] X. Song, G. Wang, Z. Wu, Y. Huang, D. Su, D. Yu, and H. Meng, "Speech-xlnet: Unsupervised acoustic model pretraining for self-attention networks," *CoRR*, vol. abs/1910.10387, 2019. [Online]. Available: http://arxiv.org/abs/1910.10387

[16] 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, pp. 5998–6008.

[17] 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, pp. 1–5.

[18] 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.

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

[20] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012. [Online]. Available: http://arxiv.org/abs/1207.0580

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016, pp. 2818–2826.

[22] Z. Tian, J. Yi, Y. Bai, J. Tao, S. Zhang, and Z. Wen, "Synchronous transformers for end-to-end speech recognition," in *ICASSP*, 2020, pp. 7884–7888.

[23] C. Shan, C. Weng, G. Wang, D. Su, M. Luo, D. Yu, and L. Xie, "Component fusion: Learning replaceable language model component for end-to-end speech recognition system," in *ICASSP*, 2019, pp. 5631–5635.

[24] Y. Bai, J. Yi, J. Tao, Z. Tian, Z. Wen, and S. Zhang, "Integrating whole context to sequence-to-sequence speech recognition," *CoRR*, vol. abs/1912.01777, 2019. [Online]. Available: http://arxiv.org/abs/1912.01777

[25] 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, pp. 1408–1412.