EXPLORING MACHINE SPEECH CHAIN FOR DOMAIN ADAPTATION

Fengpeng Yue[1*], *Yan Deng*[2], *Lei He*[2], *Tom Ko*[1†], *Yu Zhang*[1,3]

[1]Department of Computer Science and Engineering,
Southern University of Science and Technology, Shenzhen, China
[2] Microsoft China
[3] Peng Cheng Laboratory, Shenzhen, China

ABSTRACT

Machine Speech Chain integrates both end-to-end (E2E) automatic speech recognition (ASR) and neural text-to-speech (TTS) into one circle for joint training. It has been proven that it can effectively leverage a large amount of unpaired data in the spirit of data augmentation. In this paper, we explore the TTS→ASR pipeline in machine speech chain to perform domain adaptation for both E2E ASR and neural TTS models with only text data from the target domain. We conduct experiments by adapting from audiobook domain (i.e., LibriSpeech) to presentation domain (i.e., TED-LIUM). There is a relative word error rate (WER) reduction of 19.7% for the E2E ASR model on the TED-LIUM test set, and a relative WER reduction of 29.4% in synthetic speech generated by neural TTS in the presentation domain. Moreover, we observe that the gains from the proposed method and conventional adaptation methods of language models are additive.

Index Terms— machine speech chain, domain adaptation, speech recognition, speech synthesis

1. INTRODUCTION

E2E ASR [1, 2] has shown great potential and achieved state-of-the-art performance in a lot of scenarios. Also, neural approaches have become very popular in the field of TTS [3, 4]. The speech generated by state-of-the-art TTS models is hard to be distinguished from human speech. Although ASR and TTS have achieved breakthroughs in a lot of benchmarks, they still suffer from data mismatches in real-world applications. Therefore, adaptation techniques for E2E models are a popular research topic.

Text-domain mismatch is a common cause of the performance degradation in ASR and TTS. In the era of hybrid ASR, where the acoustic and language models are trained separately, text-domain adaptation can be used on the language model to improve the performance in the target domain [5]. For E2E ASR, text-domain adaptation can be achieved by fusing additional language models [6]. In parametric TTS, a method of generating optimized script for adaptation is proposed in [7]. For attention-based neural TTS, it is still challenging to learn a suitable attention mechanism for unseen text, which results in incorrect pronunciation, skipping, or repetition issues. [8] proposes a stepwise monotonic attention method to improve the robustness of TTS on the out-of-domain text.

Recently, machine speech chain [9] has drawn a lot of research interests. It provides a feasible way to train ASR and TTS models with both unpaired speech and text data. Furthermore, it can improve the two models concurrently by allowing them to teach each other. Most existing researches on machine speech chain or its variants focus on leveraging in-domain data to improve the performance of ASR. For example, [10, 11] utilize TTS on unpaired text data to generate speech as a data augmentation method. [12, 13] utilize unpaired speech by constructing a differentiable ASR→TTS pipeline or a text-to-encoder that learns to predict the hidden states of the ASR encoder. [14] uses both unpaired text and speech to update ASR by combining a differentiable ASR→TTS loss with a TTS→ASR loss. Additionally, some researches utilize machine speech machine to improve TTS. For instance, to enable TTS to speak louder in a noisy condition, [15] implements a dynamically adaptive machine speech chain inference framework by using ASR-loss embedding.

In addition to the aforementioned works, there are some studies working on using machine speech chain to help E2E ASR with domain adaptation [16, 17, 18]. In contrast to other text-domain adaptation methods [5, 6, 19, 20], where ASR can only learn from the text data, machine speech chain can also help ASR learn the corresponding acoustic representation by using TTS to generate acoustic features for the texts. However, in these works, parameters in the TTS model pre-trained in the source domain are frozen during the adaptation. Thus, TTS may suffer from domain mismatch that causes the quality degradation in synthesized speech for out-of-domain texts. Improving TTS in the target domain may further enhance the performance of ASR. This motivates us to investigate the adaptation within the machine speech chain framework where the ASR and TTS can be improved together by using only unpaired text data. Among recent works, [21] is similar to our work in which they also update the parameters of ASR and TTS concurrently. However, they only explore their work in speaker adaptation.

In this paper, we explore the machine speech chain model for the text-domain adaptation. We adopt the TTS→ASR pipeline to build a three-stage adaptation framework.

- In the first stage, we use the TTS model pre-trained on the source corpus to generate speech for the target domain to improve the ASR performance. Data filtering is used to remove unintelligible data on the fly, as the pre-trained TTS model is not robust enough for out-of-domain texts.
- In the second stage, the TTS model is adapted to the target domain by using the ASR model to provide feedback signals on the quality of synthetic features. The ASR model is fixed in this stage.
- In the third stage, we use the adapted TTS model to further improve the ASR model with more intelligible synthesized

* Initial work was done during internship at Microsoft China.
†corresponding author.

speech for out-of-domain texts.

We conduct experiments by adapting from an audiobook domain (i.e., LibriSpeech [22]) to a presentation domain (i.e., TED-LIUM [23]). Experiments show that our approach can effectively improve the performance of TTS on the target text-domain with much fewer attention errors observed. After the adaptation, there is a relative WER reduction of 19.7% for the E2E ASR model on the target domain, and a relative WER reduction of 29.4% in synthetic speech generated by neural TTS for texts from the target domain.

The rest of the paper is organized as follows. In Section 2, we review the machine speech chain framework. In Section 3, we introduce our adaptation approach. In Section 4, we describe the details of our experiments. Section 5 makes the conclusions.

2. MACHINE SPEECH CHAIN

The machine speech chain is built based on the interactive learning process of human speech perception and generation. It consists of both TTS and ASR, which are trained jointly to improve each other. The ASR task is to transcribe speech into text, similar to human speech perception. TTS is a speech generation task, which generates speech for any given text.

In the machine speech chain, attention-based E2E ASR and neural TTS are used. Given the input speech X, E2E ASR directly models the conditional probability $P(Y|X)$ between speech X and the token sequence $Y = \{y_1, y_2, ...y_T\}$ of length T as

$$p_{ASR}(Y|X) = \prod_{i}^{T} p(y_i|y_1, y_2, ...y_{i-1}, X), \tag{1}$$

where token y_i can be a character, subword or word. The cross-entropy loss used is defined as

$$\begin{aligned} L_{ASR}(Y^*, X) &= -\log p_{ASR}(Y^*|X) \\ &= -\sum_{i}^{T} \log p_{ASR}(y_i^*|y_1^*, y_2^*, ...y_{i-1}^*, X), \end{aligned} \tag{2}$$

where y_i^* denote the i-th ground-truth token.

Similar to ASR, TTS uses an attention-based E2E model. It predicts the spectrogram for input text and calculates three kinds of losses between prediction and ground truth: mean square error (MSE), mean absolute error (MAE) and stop token binary cross-entropy (BCE), with the entire loss for TTS as

$$L_{TTS} = L_{MSE} + L_{MAE} + L_{BCE}. \tag{3}$$

The MSE can be interpreted as the minus log-probability of the speech features for a Gaussian distribution $-\log p_{TTS}(X|Y)$ with a constant scale parameter [14] as

$$-\log p_{TTS}(X^*|Y) \propto ||X^* - \hat{X}||^2, \tag{4}$$

where X^* and Y denote ground truth of the spectrogram and input text token, respectively, and $\hat{X}$ is the predicted spectrogram by TTS.

There are two pipelines in machine speech chain. One pipeline is TTS→ASR to leverage unpaired text and the other is ASR→TTS to leverage unpaired speech. Only TTS→ASR pipeline is needed for domain adaptation when only text data is available.

In the TTS→ASR pipeline, TTS generates speech spectrogram $\hat{X}$ for given unpaired text Y as

$$\hat{X} = \arg\max_{X} \{p_{TTS}(X|Y)\}. \tag{5}$$

Then Y and $\hat{X}$ are used to calculate $L_{TTS\rightarrow ASR}$ with the ASR model as

$$L_{TTS\rightarrow ASR} = L_{ASR}(\hat{X}, Y). \tag{6}$$

There have been some studies on using TTS to do data augmentation to improve the performance of ASR. For example, in [10], TTS and ASR are trained separately, which can avoid the effort of training new models based on the same acoustic feature for joint training. Its experiment verified that even for the ASR model that is trained on 960 hours of data with SpecAugment [24], there is still slight improvement after adding synthetic data generated by TTS. With the help of a consistency loss between ASR prediction on real and synthetic speech, the paired training data for ASR can be reduced by half with little performance degradation [11]. In addition to using the TTS→ASR pipeline to improve the performance of ASR, [25] uses it to resolve the robustness issue for neural TTS, which is induced by using a reference with a totally different content for style transfer.

3. ADAPTATION FRAMEWORK

Most domain adaptation methods usually improve the ASR or TTS by using either unpaired speech or text. In contrast, we explore updating parameters in the ASR and TTS models concurrently in the TTS→ASR pipeline. [1] We first pretrain the ASR and TTS models on the source domain data, and then use the TTS→ASR pipeline to perform three-stage training for domain adaptation as shown in Algorithm 1. For each unpaired text, we randomly select a speaker in the training corpus from the source domain to generate acoustic features.

Firstly, We use the TTS model trained on corpus from the source domain to generate speech for text from the target domain. For attention-based neural TTS models, it has been proven that there may be a few attention issues in generated samples, especially for out-of-domain texts [8]. This kind of generated data with attention issues can be considered as outliers , leading to the performance degradation if we use them to fine-tune the ASR model. Here, we use the focus rate [4] to filter such outliers. The focus rate F measures how an attention head is close to diagonal: $F = \frac{1}{S}\sum_{s=1}^{S} max_{1\leq t\leq T}\alpha_{s,t}$, where S and T are the lengths of the generated spectrograms and phonemes, $\alpha_{s,t}$ donates the element in the s-th row and t-th column of the attention matrix. The threshold of the focus rate is set based on a validation set, and in our experiments, it is set to 0.58.

In the second stage, we use the fine-tuned ASR model to act as a discriminator to provide feedback signals on the acoustic features generated by the TTS model. Data filtering is not applied at this stage, as we expect to improve the robustness of the TTS model in the target domain after training. In the experiment, we use $\alpha = 0.005$ that is set based on a validation set in Algorithm 1 for the TTS→ASR loss of unpaired text from target domain.

Finally, we use the adapted TTS with improved intelligibility to further fine-tune ASR through TTS→ASR pipeline with an expectation to obtain more gains for the ASR model in the target domain. In our experiments, there is no more gain for both TTS and ASR models after three iterations, so we only use three-stage training here.

[1] Our implementation, which is based on ESPnet [26], is available at `https://github.com/fengpeng-yue/ASRTTS`.

Algorithm 1 TTS $\rightarrow$ ASR Adaptation

Input: paired data of ASR source domain D^{AS}, paired data of TTS source domain D^{TS}, unpaired text data of target domain D^{T}, pre-trained ASR_{init}, pre-trained TTS_{init}, the threshold for the focus rate γ, the coefficient α for the TTS adaptation loss

Stage 1:

1: Sample paired data from source domain D^{AS}
$\left(x^{AS}, y^{AS}\right) = \left([x_1^{AS}, ...x_N^{AS}], [y_1^{AS}, ...y_N^{AS}]\right)$
2: Sample text $y^T = \left[y_1^T, ...y_M^T\right]$ from the target domain D^T
3: Generate speech by TTS_{init}:
$\hat{x}^T \sim P_{TTS_{init}}\left(\cdot|y^T; \Theta_{TTS_{init}}\right)$
4: Filter data according to the threshold γ:
$\left(\overline{x}^T, \overline{y}^T\right) = filter(\hat{x}^T, y^T, \gamma)$
5: Calculate loss for ASR:
$l_{ASR} = L_{ASR_{init}}(x^{AS}, y^{AS}) + L_{ASR_{init}}(\overline{x}^T, \overline{y}^T)$
6: Update ASR_{init} to ASR_{iter1}

Stage 2:

7: Sample paired data from source domain D^{TS}
$\left(x^{TS}, y^{TS}\right) = \left([x_1^{TS}, ...x_N^{TS}], [y_1^{TS}, ...y_N^{TS}]\right)$
8: Sample text $y^T = \left[y_1^T, ...y_M^T\right]$ from the target domain D^T
9: Generate speech by TTS_{init}:
$\hat{x}^T \sim P_{TTS_{init}}\left(\cdot|y^T; \Theta_{TTS_{init}}\right)$
10: Calculate the loss for TTS:
$l_{TTS} = L_{TTS\rightarrow ASR}(\hat{x}^T, y^T) + \alpha L_{TTS_{init}}(x^{TS}, y^{TS})$
11: Update TTS_{init} to TTS_{adapt}

Stage 3:

12: Replace ASR_{init} with ASR_{iter1},
Replace TTS_{init} with TTS_{adapt} and repeat the stage 1
13: Update ASR_{iter1} to ASR_{adapt}

Output: ASR_{adapt}, TTS_{adapt}

4. EXPERIMENTS AND RESULTS

4.1. Experimental Setup

4.1.1. Dataset

We use LibriSpeech [22] as the ASR source domain dataset and LribriTTS [27] as the TTS source domain dataset. We pretrain the ASR baseline model on LibriSpeech train-clean-460 set and pretrain the TTS baseline model on LibriTTS train-clean-460 set. To make the spectrogram synthesized by TTS in the adaptation consistent with the sampling frequency of the pre-trained ASR model, we down-sample LibriTTS to 16kHZ. We use 80-dimensions Mel-filter bank coefficients (FBANK) features computed on 50ms windows with 10ms shift. The window size follows the common setting in the TTS model. The SpecAugment [24] is used for both ASR baseline and the adaptation to improve the performance. We use the byte pair encoder (BPE) [28] as ASR token units and phoneme as TTS token units.

The text contents of LiriSpeech and LibriTTS come from audio-book, which are in a written style with a formal wording and strict grammar. TED-LIUM texts come from presentations, which are in a spoken style and more casual grammatically. Therefore, we choose TED-LIUM as the target domain and use all texts from training set for the TTS$\rightarrow$ASR pipeline. To match the pre-trained TTS model, we add punctuations for TED-LIUM texts automatically.

For the language model, we use texts from LibriSpeech-lm-norm and LibriSpeech training set as the source domain data, and all texts from TED-LIUM as the target domain data. The details of each dataset are shown in Table 1.

Table 1: The statistics of the datasets used.

	#Sentence	#Token
Source domain	40M	102M
Target domain training set	58K	170K
Target domain test set	1155	35K

4.1.2. Model Configuration

In our experiments, we use listen, attend and spell (LAS [2]) as the ASR model. The encoder includes two convolutional layers and four bidirectional LSTM (BLSTM) layers. The first and second convolutional layers contain 64 and 128 filters with shape 3 x 3 and a 1x 1 stride, respectively. The max-pooling layer following each convolutional layer downsample the size of the input to half. The decoder contains two unidirectional LSTM layers with 1024 units. We use the location-based attention [1] to improve the performance of ASR. The AdaDelta optimizer is used to train ASR and the threshold of gradient clipping is set to 5. For attention-based models, there are sometimes a few insertions or deletions. To improve the robustness of the decoding, we restrict the length ratio of hypothesis to input speech to be within [0.1, 0.6]. The size of the beam search is set to 10.

For the TTS model, we use Tacotron2 [3, 29] as the basic network architecture, and use a pre-trained speaker model to extract speaker embedding for the multi-speaker TTS modeling. The parameters of the encoder, attention and decoder are the same as the original paper unless otherwise stated. The dimension of the speaker embedding is set to 512. We use the Adam optimizer and the threshold of gradient clipping is set to 1. To get more stable alignments on complex corpus, we adopt the guided attention loss [30] during TTS model training.

Adapting a pre-trained language model (LM) using the larger transformer model to the target domain has been proven simple and effective [19]. We also compare the proposed method with the LM adaptation by training two different LMs: 1) LSTM-LM trained on the target domain training set, which has 2 LSTM layers with 800 nodes in each layer; 2) a transformer LM (Trans-LM) pre-trained with the source domain data and then fine-tuned on the target domain data. Trans-LM has 12 layers with 512 hidden units in each layer and 8 heads in each multi-head attention block. We used the Adam optimizer and set the learning rate to 10^{-6} during the fine-tuning.

4.2. Results and Discussions

4.2.1. Domain Adaptation for ASR

In this experiment, the baseline LAS model is trained on LibriSpeech. The word error rate (WER) is 7% on the LibriSpeech test set, which is comparable with previous studies that train models by using 460h train-clean set only. When we evaluate this model on TED-LIUM, there is a significant regression in performance. The WER becomes 27.8 % on the test set and 30 % on the dev set, as shown in Table 2.

Then, we try the first stage of the proposed method to perform domain adaptation by using Tacotron2. The results are shown in the

Table 2: Performance of domain adaptation for LAS.

Method	WER(%)	
	TED test set	TED dev set
Baseline	27.8	30.0
+TTS(w/o adaptation)	24.0	25.3
+SpecAugment	23.6	23.9
+Data filtering	23.1	23.6
+TTS(w/ adaptation)	22.3	22.2

second row of Table 2 with a reduction in WER to 24% on the test set and 25.3% on the dev set. Further, we add SpecAugment and slight more gain can be obtained on both test and dev sets. As there are always a few samples with unintelligible segments generated by Tacotron2, we try to add data filtering during training, leading to a slight WER reduction, as shown in the fourth row of Table 2. To obtain further improvements through the proposed method, we apply the whole three-stage training strategy as described in Section 3. The results in the last row of Table 2 show the effectiveness of the proposed method and training strategy. The final WER can be reduced to 22.3% on the test set and 22.2% on the dev set. Overall, we achieve a relative reduction of 19.7% on the TED-LIUM test set over a well-trained baseline.

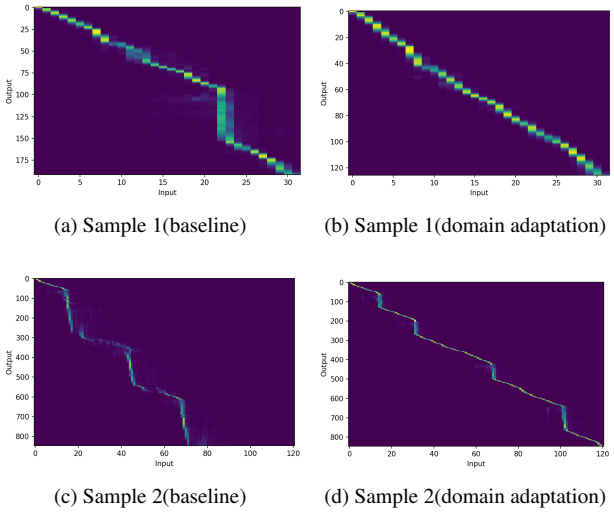
We also compare the proposed method with LM shallow fusion, and try to combine them to see if they complement each other. The perplexity of the LSTM-LM and Trans-LM described in Section 4.1.2 is 62.7 and 39.0 on the TED-LIUM test set, respectively. The fine-tuned Trans-LM performs much better than the LSTM-LM that is only trained on the target domain data. All decoding results are shown in Table 3. We can see that the proposed method is more effective than LM shallow fusion with different LMs, and further improvements can be obtained when using LM shallow fusion on top of the proposed method. The final WER is reduced to 21.4% on the TED-LIUM test set.

Table 3: Shallow fusion with original and adapted LM

Method	WER(%)	
	TED test set	TED dev set
Baseline	27.8	30.0
+LSTM-LM	26.7	29.6
+Trans-LM(fine-tuned)	26.4	29.7
Our method	22.3	22.2
+LSTM-LM	21.9	21.8
+Trans-LM(fine-tuned)	21.4	21.6

4.2.2. Domain Adaptation for TTS

The TTS model is also updated at the second stage of the proposed training strategy. We show the improvements on Tacotron2 in Table 4. Here, the baseline LAS model before the adaptation is used to evaluate the synthesized speech by Tacotron2. According to Table 4, we can see that significant improvements (i.e., about 29.4% relative WER reduction in generated samples) are obtained in Tacotron2 for input texts from the target domain (i.e., the TED-LIUM test set). Also, the stability of Tacotron2 for texts from the source domain is improved in that the WER of synthesized speech is reduced by about 10.6% relatively on the LibriSpeech test set. Based on the results,



(a) Sample 1(baseline) (b) Sample 1(domain adaptation)

(c) Sample 2(baseline) (d) Sample 2(domain adaptation)

Fig. 1: Alignments of generated samples for out-of-domain texts. The horizontal axis represents encoder timesteps, and the vertical axis represents decoder timesteps.

we can see that the proposed method is also effective in improving the intelligibility of attention-based neural TTS.[2]

Table 4: Performance of domain adaptation for the Tacotron2. The texts and reference utterances are randomly matched to generate speech examples.

Method	WER(%)	
	LibriSpeech test set	TED test set
Baseline	12.2	22.4
+Domain adaptation	10.9	15.8

To further understand the improvements in the robustness, we plot in Figure 1 the alignment of the input text and the output Mel-spectrogram generated by Tacotron2. According to Figure 1, we can see that the alignment of samples from the adapted model seems much better than baseline.

5. CONCLUSIONS

In this paper, we study domain adaptation in the TTS→ASR pipeline of machine speech chain and propose the three-stage training strategy. Experimental results show that significant improvements (i.e., about 19.7% reduction in WER for ASR and 29.4% for attention-based neural TTS) can be obtained for ASR and TTS in a different domain. We also verify the effectiveness of data filtering and adaptation of TTS. Moreover, we find that the proposed method is more effective than LM shallow fusion in the adaptation of ASR model, as more acoustics can be explored. Furthermore, the proposed method is complementary to LM shallow fusion.

[2]Samples of synthesized speech can be found in `https://fengpeng-yue.github.io/ASRTTS_adapt`

6. REFERENCES

[1] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, "Attention-based models for speech recognition," in *Proc. NIPS*, 2015.

[2] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016.

[3] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al., "Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions," in *Proc. ICASSP*, 2018.

[4] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu, "Fastspeech: Fast, robust and controllable text to speech," in *Proc. NIPS*, 2019.

[5] Horia Cucu, Laurent Besacier, Corneliu Burileanu, and Andi Buzo, "ASR domain adaptation methods for low-resourced languages: Application to Romanian language," in *Proc. EUSIPCO*, 2012.

[6] Changhao Shan, Chao Weng, Guangsen Wang, Dan Su, Min Luo, Dong Yu, and Lei Xie, "Component fusion: Learning replaceable language model component for end-to-end speech recognition system," in *Proc. ICASSP*, 2019.

[7] Min Chu, Chun Li, Hu Peng, and Eric Chang, "Domain adaptation for TTS systems," in *Proc. ICASSP*, 2002.

[8] Mutian He, Yan Deng, and Lei He, "Robust Sequence-to-Sequence Acoustic Modeling with Stepwise Monotonic Attention for Neural TTS," in *Proc. Interspeech*, 2019.

[9] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura, "Listening while speaking: Speech chain by deep learning," in *Proc. ASRU*, 2017.

[10] Nick Rossenbach, Albert Zeyer, Ralf Schlüter, and Hermann Ney, "Generating synthetic audio data for attention-based speech recognition systems," in *Proc. ICASSP*, 2020.

[11] Gary Wang, Andrew Rosenberg, Zhehuai Chen, Yu Zhang, Bhuvana Ramabhadran, Yonghui Wu, and Pedro Moreno, "Improving speech recognition using consistent predictions on synthesized speech," in *Proc. ICASSP*, 2020.

[12] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura, "End-to-end feedback loss in speech chain framework via straight-through estimator," in *Proc. ICASSP*, 2019.

[13] Tomoki Hayashi, Shinji Watanabe, Yu Zhang, Tomoki Toda, Takaaki Hori, Ramon Astudillo, and Kazuya Takeda, "Back-translation-style data augmentation for end-to-end ASR," in *Proc. SLT*, 2018.

[14] Murali Karthick Baskar, Shinji Watanabe, Ramon Astudillo, Takaaki Hori, Lukáš Burget, and Jan Černocký, "Semi-Supervised Sequence-to-Sequence ASR Using Unpaired Speech and Text," in *Proc. Interspeech*, 2019.

[15] Sashi Novitasari, Sakriani Sakti, and Satoshi Nakamura, "Dynamically Adaptive Machine Speech Chain Inference for TTS in Noisy Environment: Listen and Speak Louder," *Proc. Interspeech*, 2021.

[16] Jinyu Li, Rui Zhao, Zhong Meng, Yanqing Liu, Wenning Wei, Sarangarajan Parthasarathy, Vadim Mazalov, Zhenghao Wang, Lei He, Sheng Zhao, and Yifan Gong, "Developing RNN-T Models Surpassing High-Performance Hybrid Models with Customization Capability," in *Proc. Interspeech*, 2020.

[17] Xianrui Zheng, Yulan Liu, Deniz Gunceler, and Daniel Willett, "Using synthetic audio to improve the recognition of out-of-vocabulary words in end-to-end ASR systems," in *Proc. ICASSP*, 2021.

[18] Murali Karthick Baskar, Lukáš Burget, Shinji Watanabe, Ramon Fernandez Astudillo, et al., "Eat: Enhanced ASR-TTS for Self-Supervised Speech Recognition," in *Proc. ICASSP*, 2021.

[19] Ke Li, Zhe Liu, Tianxing He, Hongzhao Huang, Fuchun Peng, Daniel Povey, and Sanjeev Khudanpur, "An empirical study of transformer-based neural language model adaptation," in *Proc. ICASSP*, 2020.

[20] Janne Pylkkönen, Antti Ukkonen, Juho Kilpikoski, Samu Tamminen, and Hannes Heikinheimo, "Fast Text-Only Domain Adaptation of RNN-Transducer Prediction Network," in *Proc. Interspeech*, 2021.

[21] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura, "Machine Speech Chain with One-shot Speaker Adaptation," in *Proc. Interspeech*, 2018.

[22] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015.

[23] Anthony Rousseau, Paul Deléglise, and Yannick Esteve, "TED-LIUM: an Automatic Speech Recognition dedicated corpus.," in *Proc. LREC*, 2012.

[24] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech*, 2019.

[25] Da-Rong Liu, Chi-Yu Yang, Szu-Lin Wu, and Hung-Yi Lee, "Improving unsupervised style transfer in end-to-end speech synthesis with end-to-end speech recognition," in *Proc. SLT*, 2018.

[26] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai, "ESPnet: End-to-End Speech Processing Toolkit," in *Proc. Interspeech*, 2018.

[27] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu, "LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech," in *Proc. Interspeech*, 2019.

[28] Rico Sennrich, Barry Haddow, and Alexandra Birch, "Neural machine translation of rare words with subword units," in *Proc. ACL*, 2016.

[29] Ye Jia, Yu Zhang, Ron J Weiss, Quan Wang, Jonathan Shen, Fei Ren, Zhifeng Chen, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, et al., "Transfer learning from speaker verification to multispeaker text-to-speech synthesis," in *Proc. NIPS*, 2018.

[30] Hideyuki Tachibana, Katsuya Uenoyama, and Shunsuke Aihara, "Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention," in *Proc. ICASSP*, 2018.