ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-6654-0540-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICASSP43922.2022.9746217

SYNT++: UTILIZING IMPERFECT SYNTHETIC DATA TO IMPROVE SPEECH RECOGNITION

Ting-Yao Hu[⋆] *Mohammadreza Armandpour*[†] *Ashish Shrivastava*[⋆]
Jen-Hao Rick Chang[⋆] *Hema Koppula*[⋆] *Oncel Tuzel*[⋆]

[⋆]Apple [†]Texas A&M University

ABSTRACT

With recent advances in speech synthesis, synthetic data is becoming a viable alternative to real data for training speech recognition models. However, machine learning with synthetic data is not trivial due to the gap between the synthetic and the real data distributions. Synthetic datasets may contain artifacts that do not exist in real data such as structured noise, content errors, or unrealistic speaking styles. Moreover, the synthesis process may introduce a bias due to uneven sampling of the data manifold. We propose two novel techniques during training to mitigate the problems due to the distribution gap: (i) a rejection sampling algorithm and (ii) using separate batch normalization statistics for the real and the synthetic samples. We show that these methods significantly improve the training of speech recognition models using synthetic data. We evaluate the proposed approach on keyword detection and Automatic Speech Recognition (ASR) tasks, and observe up to 18% and 13% relative error reduction, respectively, compared to naively using the synthetic data.

Index Terms— Data augmentation, speech recognition, synthetic data

1. INTRODUCTION

Collecting and annotating datasets for training speech recognition models is hard and expensive. An alternative approach is to generate a synthetic dataset using speech synthesis models [1–5]. However, training recognition models with synthetic data is not trivial. Figure 1 shows an illustration of the joint distribution of speech signals and corresponding text labels – black is the true data distribution and green is the synthetic data distribution. In general, we do not have access to the true data distribution, but have finite sample approximation through a collected real dataset. Although the quality of synthetic speech (synthesized via a controllable generative model) has improved significantly, there is still a gap between the synthetic and the true data distributions. We can characterize this gap into 4 regions as shown in Figure 1: (i) artifacts (e.g. structured noise, speech/content mismatch, unrealistic styles in synthetic data) where the synthesized samples are outside the support of true distribution, (ii) over-sampled region where the synthetic distribution has more mass and (iii) under-sampled region where the synthetic distribution has less mass compared to the true distribution, and (iv) missing samples where the synthetic distribution has zero mass inside the support of the true distribution (e.g. missing speaking styles, accents).

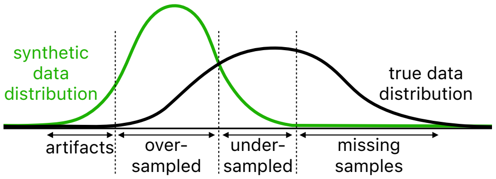

Fig. 1: The joint distributions of speech and corresponding text. The gap between synthetic and true data distributions can be partitioned into 4 regions. See the first paragraph of the introduction section for details.

In this paper, we present Synt++, an improved algorithm to utilize synthetic data for training speech recognition models. Synt++ combines two novel techniques. The first technique is a rejection sampling algorithm [6–8] that modifies the sampling process of the synthesis model (the generative model) to make the synthesized data distribution closer to the true data distribution. Specifically, we train a discriminator function to classify samples as real or synthetic. We then use the probability of real measure predicted by this discriminator to stochastically accept/reject synthesized samples. The rejection sampling addresses the first three problems discussed above by rejecting artifacts, and under/over sampling data points in over/under-sampled regions, respectively. Note that the rejection sampling cannot correct for the missing samples since the synthesis model does not have support and, hence, cannot synthesize the samples in this region.

The second technique accounts for the remaining gap during training by using separate Batch Normalization (BN) statistics for real and synthetic data. BN is a commonly used technique to normalize the features during training to address covariate shift and improve convergence. However, the BN statistics estimated using a combination of synthetic and real data becomes biased due to the distribution gap (e.g., in Figure 1, the mean of the combined real/synthetic distribution is shifted left and the variance is larger than the true distribution).

[†]Work done during summer internship at Apple. Emails: [⋆]{tingyao_hu, ashish.s, jenhao_chang, hkoppula, otuzel}@apple.com, [†]armand@stat.tamu.edu

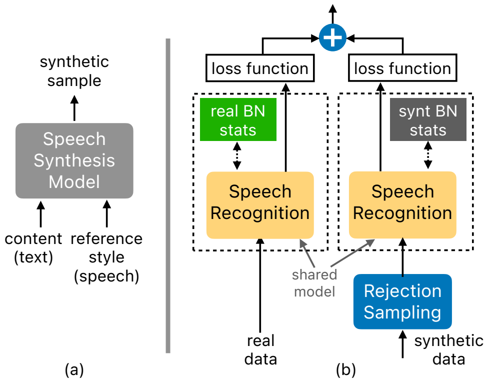

Fig. 2: Method overview. (a) To synthesize data, we use a controllable generative model [5] that takes a text and a reference speech as input and utters the text in the style of the given speech. (b) During training of the recognition model, we use rejection sampling and separate BN statistics to address the distribution gap between the real and the synthetic data.

To prevent this problem, we estimate real and synthetic data BN statistics separately, and utilize only the real BN statistics during inference. Moreover, this process helps to bridge the domain gap since synthetic and real data are normalized separately to have similar statistics.

2. RELATED WORK

State-of-the-art neural Text-to-Speech (TTS) architectures, such as Tacotron 2 [9] and FastSpeech 2 [10], are capable of generating speech indistinguishable from human speech. Controllable speech synthesis extends the TTS models beyond modeling the voice of a single or a few speakers to the entire style space of speech [3–5, 11]. Using controllable synthesis models, we can expand the acoustic variations of synthetic speech or use a bigger text corpus to add data for unseen content.

Previous works have used TTS models to improve the training of speech recognition models. One group of work relies on the TTS models and self/semi-supervised learning techniques to exploit unpaired speech and text data in the ASR training process [12–16]. Another group leverages TTS models to create more training data when the amount of real data is limited [17–21]. Some other works have focused on specific tasks, such as out-of-vocabulary word recognition [22], and keyword detection [23, 24]. These methods, however, do not address the imperfections of the speech generated by the synthesis models.

The idea of utilizing synthetic data to improve a recognition model has also been explored in the computer vision domain. In [25], the authors proposed to refine synthetic images via adversarial training to bridge the distribution gap. In [26], the authors showed that a generative model could be trained to augment images of unseen classes. In [27], authors use adversarial examples to improve image recognition.

3. METHOD

3.1. Controllable Speech Generative Model

Controllable speech generative models can generate speech in a given style, where the content is specified by an input text. As shown in Figure 1, for effective learning with synthetic data, we need a generative model that has *as small gap as possible* with the true distribution. Therefore, the generative model should be able to synthesize any text in a wide range of speaking styles in various environmental conditions such as background noise, microphone response, etc. without introducing artifacts to the signal. To this end, we utilize a recently introduced auto regressive controllable generative model called Style Equalization [5] that enables learning a controllable generative model in the wild (thousands of speakers recorded in various conditions) without requiring style labels. As shown in Figure 2(a), the model takes a text and a reference style speech as input and utters the text in the style of the given reference. In this model, the style is broadly defined, not only the speaker's voice characteristics but also all aspects of the reference speech sample, including background noise, echo, microphone response, etc., which are necessary to have a smaller distribution gap (particularly missing samples). This model also allows sampling styles from a prior distribution that is important for sampling new styles not contained in the dataset. We refer readers to [5] for a detailed description of this model.

3.2. Rejection Sampling

Let $\boldsymbol{x}$ be a speech signal and y be the corresponding token label, and $p_d(\boldsymbol{x}, y)$, $p_g(\boldsymbol{x}, y)$ denote the joint true and synthetic data distributions, respectively. We use the synthesizer as proposal distribution and use rejection sampling to make it closer to the true distribution. Following standard rejection sampling [6], a synthetic data $(\boldsymbol{x}, y)$ is accepted with probability $\frac{p_d(\boldsymbol{x},y)}{Mp_g(\boldsymbol{x},y)}$, where scalar $M > 0$ is selected such that $Mp_g(\boldsymbol{x}, y) > p_d(\boldsymbol{x}, y), \forall(\boldsymbol{x}, y)$ in domain of p_d, i.e., $M = \max_{\boldsymbol{x},y} \frac{p_d(\boldsymbol{x},y)}{p_g(\boldsymbol{x},y)}$. Azadi et. al. [7] proposed a rejection sampling method to match the distribution of the images from a generative adversarial network to the true distribution; however, their algorithm does not consider the joint distribution of $(\boldsymbol{x}, y)$ that is needed to train downstream recognizers.

To approximate the ratio $\frac{p_d(\boldsymbol{x},y)}{p_g(\boldsymbol{x},y)}$, we train a discriminator, $D(\boldsymbol{x}, y)$, that takes a speech signal and the corresponding text and produces probability of $(\boldsymbol{x}, y)$ coming from true distribution, i.e., $D(\boldsymbol{x}, y) \in [0, 1]$. When D is trained to its optimal value D^*, it satisfies ([7, 8, 28])

$$D^*(\boldsymbol{x}, y) = \frac{p_d(\boldsymbol{x}, y)}{p_d(\boldsymbol{x}, y) + p_g(\boldsymbol{x}, y)}, \tag{1}$$

which can be re-arranged to yield

$$r(\boldsymbol{x}, y) := \frac{p_d(\boldsymbol{x}, y)}{p_g(\boldsymbol{x}, y)} = \frac{D^*(\boldsymbol{x}, y)}{1 - D^*(\boldsymbol{x}, y)}. \tag{2}$$

To train D, we minimize

$$\mathscr{L}_D = \mathbb{E}_{(\boldsymbol{x},y)\sim p_d}(\log(D(\boldsymbol{x},y))) + \mathbb{E}_{(\boldsymbol{x},y)\sim p_g}(1-\log(D(\boldsymbol{x},y))),$$

which is equivalent to training a two class classification model, where the positive class (label 1) corresponds to the real samples and the negative class (label 0) corresponds to the synthetic samples. Note that, we model the joint speech and label distribution, therefore D should also utilize whether the speech signal matches the token labels. To this end, we use a pre-trained ASR model, A to predict the token labels, $\hat{y}$, of the speech signal, i.e., $\hat{y} = A(\boldsymbol{x})$. Then, we characterize the discrepancy between $\hat{y}$ and y by a 5-dimensional feature vector $\Phi(\boldsymbol{x},y)$, whose elements include the cross-entropy (CE) loss, connectionist temporal classification loss, word error rate (WER), and number of tokens in y and $\hat{y}$. Finally, we train a two-layer DNN, $D_{\boldsymbol{\theta}}$, that takes $\Phi(\boldsymbol{x},y)$ as input and produces the probability of the sample being real

$$D(\boldsymbol{x},y) := D_{\boldsymbol{\theta}}(\Phi(\boldsymbol{x},y)), \tag{3}$$

where $\boldsymbol{\theta}$ denotes the parameters of the DNN.

Practically, we need to compute M and address the fact that the rejection sampling can be slow to select a fixed number of samples. First, we estimate an initial value of M by computing the maximum value of p_d/p_g (using D^* and Eq. 2) over a few (approximately 200) generated samples. Following [7], to compute the acceptance probability for each generated sample, $(\boldsymbol{x},y)$, we first compute the discriminator score $D^*(\boldsymbol{x},y)$ and then set $M \leftarrow \max(M, p_d(\boldsymbol{x},y)/p_g(\boldsymbol{x},y)) = \max(M, \frac{D^*(\boldsymbol{x},y)}{1-D^*(\boldsymbol{x},y)}))$. We stop the sampling process after accepting N synthetic samples. Note that, the proposed rejection sampling technique applies only to the domain where the generative model has support and ignores the missing samples region shown in Figure 1.

3.3. Double Batch Normalization Statistics

A BN layer [29] computes the mean and the standard deviation of the intermediate features and normalizes these features by subtracting the mean and dividing by the standard deviation within a batch. During training, the model uses means and variances computed over the samples in a mini-batch. A running average of the BN statistics, which we call running statistics, is computed to be used during inference.

Naively using BN will update the running statistics with samples from both the real and the synthetic distributions and, due to the distribution gap, the estimates will be biased (i.e. the estimated BN statistics will not match the true data distribution). Instead, we form the mini-batches consisting of only the real samples or only the synthetic samples, and compute two sets of running statistics – one from the real mini-batches and the other from the synthetic mini-batches. Since the test data (during inference) consists of only real samples, we discard the synthetic statistics and use only the real statistics. Note that, we share the affine parameters of the BN between synthetic and real batches.

The proposed technique is similar to Xie et al. [27] that used auxiliary BN layers in an adversarial learning framework where images augmented with adversarial perturbations were used to improve model accuracy. In contrast, we use separate BN statistics to improve model training with synthetic data.

4. EXPERIMENTS

4.1. Automatic Speech Recognition

For ASR experiments, we use the LibriSpeech dataset [30], which contains $1,000$ hours of speech from public domain audiobooks. Following the standard protocol, we evaluate our method on the full training set (LibriSpeech 960h) and a subset containing clean speech with only US English accents (LibriSpeech 100h). To generate synthetic datasets, we use Style Equalization [5] that is trained with LibriTTS dataset [31]. LibriTTS is a subset of LibriSpeech dataset and contains 555 hours of speech data. To study the effect of model size on synthetic data training, we train two ASR models – a large model with 116 million parameters (Large 116M) and a smaller model with 22 million parameters (Regular 22M)

ASR model: We use the implementation from ESPNet [32]) for an end-to-end ASR model. The ASR model is composed of a conformer-based encoder [33] and a transformer-based decoder [34]. Setting the encoder dimension to 144 and 512, we construct the regular (22M) and the large (116M) models. We apply SpecAugment [35] to all speech samples to further enhance the acoustic diversity. The model checkpoint of each epoch is saved, and the final model is produced by averaging the 10 checkpoints with the best validation accuracy. All ASR models are evaluated without a language model.

Baselines: For synthetic-only training, we compare naively using synthetic data with the proposed rejection sampling (Section 3.2). Note that when the training data consists of only synthetic samples, we do not have any real data to compute the running BN statistics. Hence, we do not use the double BN when training only with synthetic data. For joint (real,synt) training, our baseline is naively adding the synthetic data to training. We compare this baseline with the proposed method (real, synt++) where we use the rejection sampling algorithm described in Section 3.2 and also use separate BN statistics for the real and the synthetic samples (Section 3.3).

Synthetic data with the same text corpus as the real data: First, we study the effectiveness of the synthetic data when the synthetic samples are used to increase the acoustic variation in the training data. To this end, we use the same training corpus as the real data in LibriSpeech 960h. Specifically, for each sentence y_i in the training dataset, we take a random real training sample, $\boldsymbol{x}_j$, as the style reference, and generate synthetic sample $\boldsymbol{x}'_i$. For both Synt and Synt++ (using rejection sampling), we obtain a synthetic dataset containing 960 hours of speech.

	Real	Synt	Synt++	Real, Synt	Real, Synt++
Regular (22M)					
test-clean	3.7 ± 0.14	7.3 ± 0.17	7.0 ± 0.05	3.4 ± 0.05	**3.2** ± 0.08
test-other	9.5 ± 0.12	21.0 ± 0.12	20.0 ± 0.17	9.3 ± 0.21	**8.5** ± 0.05
Large (116M)					
test-clean	2.9 ± 0.05	6.3 ± 0.05	5.4 ± 0.12	3.0 ± 0.05	**2.6** ± 0.05
test-other	6.9 ± 0.08	18.2 ± 0.17	17.2 ± 0.17	7.4 ± 0.17	**6.5** ± 0.05

Table 1: ASR results on LibriSpeech 960h dataset. The reported numbers are percentage WER (lower is better). Synt++ significantly improves training with synthetic data.

	LibriSpeech 100h			LibriSpeech 960h		
	Real	Real, Synt	Real, Synt++	Real	Real, Synt	Real, Synt++
test-clean	7.7 ± 0.08	4.3 ± 0.08	**4.0** ± 0.08	2.9 ± 0.05	2.7 ± 0.08	**2.4** ± 0.05
test-other	20.9 ± 0.64	13.2 ± 0.21	**13.2** ± 0.08	6.9 ± 0.08	7.0 ± 0.25	**6.3** ± 0.05

Table 2: Effect of using an extended text corpus.

The WERs of the recognition models are reported in Table 1 on the two official test sets (test-clean and test-other). Synt++ gives significant improvement over naively adding synthetic data to the training (Synt).

Synthetic data with extended corpus: We study the effect of synthesizing speech from an extended text corpus in Table 2. In addition to LibriSpeech 960h, we also report the results when we have a smaller real dataset (LibriSpeech 100h). For LibriSpeech 100h, we use the text corpus of LibriSpeech 960h and synthesize 960 hours of synthetic data. For LibriSpeech 960h dataset, we double its text corpus by adding the text from the language model corpus of LibriSpeech and synthesize 960 hours of synthetic data. In both cases, extended corpus significantly improves the performance. Particularly, for the smaller dataset, we observe 48% relative reduction in WER (7.7% to 4.0%), compared to real-only training.

Ablation study: To isolate the effect of the rejection sampling and the double BN, we conduct an ablation study in Table 3 on LibriSpeech 960h dataset. As seen in this table, both the rejection sampling and the double BN statistics are important to effectively utilize the synthetic samples.

4.2. Keyword Detection

We also evaluate the proposed approach on keyword detection task using the Speech Command dataset [36]. This dataset consists of 35 keywords from multiple speakers, and samples from various background noises. For each keyword, we split the train and test subset based on the speaker identities, i.e. training and test samples did not have common speakers. We chose 3 keywords (‘down’, ‘no’, ‘stop’) to study the improvements with synthetic data. Each of these keywords contains approximately 2500 training samples and 800 test samples.

	Regular (22M)		Large (116M)	
	test-clean	test-other	test-clean	test-other
Real, Synt	3.4 ± 0.05	9.3 ± 0.21	3.0 ± 0.05	7.4 ± 0.17
+ rejection	3.4 ± 0.05	9.0 ± 0.12	2.8 ± 0.02	7.0 ± 0.05
+ dbl BN	3.3 ± 0.02	8.5 ± 0.02	2.7 ± 0.05	6.7 ± 0.02
+ rej. + dbl BN	**3.2** ± 0.08	**8.5** ± 0.05	**2.6** ± 0.05	**6.5** ± 0.05

Table 3: Ablation study on the ASR task with LibriSpeech 960h dataset.

keywords	Real	Synt	Synt++	Real, Synt	Real, Synt++
‘down’	9.6 ± 0.8	11.2 ± 0.1	8.9 ± 0.9	5.6 ± 0.2	**5.0** ± 0.2
‘no’	13.5 ± 0.8	14.5 ± 0.7	11.0 ± 0.2	7.8 ± 0.4	**6.4** ± 0.4
‘stop’	3.5 ± 0.3	11.2 ± 0.6	9.6 ± 0.4	2.1 ± 0.3	**1.5** ± 0.1

Table 4: Keyword detection results on Speech Command dataset. The reported numbers are average false accept rate in percentage (lower is better).

We set up the keyword detection task as a 2-class classification problem, where the positive class is one of the keywords and the negative class contains all the other 34 keywords and the background noise. Each sample consists of approximately 1 second of audio, which is converted to a mel-spectrogram before giving it as input to a ResNet model with approximately 6k parameters. The model is trained with cross-entropy loss.

Sampling synthetic data: For each keyword, we generated 10k samples by randomly selecting the reference styles from the real dataset. For the positive class, we generated an additional 100k samples. For both synthetic and real training, we added approximately 5k samples from background noise to the negative class. Since the detection model predicts the keyword probability (unlike a token sequence in the ASR task), we use only the CE loss in Equation 3 to compute D^*.

Metric: Since keyword detection is a 2-class problem, we can compute false reject rate (FRR) and false accept rate (FAR) at different thresholds. The detection error tradeoff (DET) curve is commonly used for keyword detection tasks [37]. Since lower FRR is preferred for keyword detection, we report average FAR for FRR values over the range of 0-5%, which is equivalent to the area under the curve (AUC) of the DET curve over the FRR range of 0-5%.

Results: We report the average FAR for keywords in Table 4 and observe significant improvements of the proposed approach, compared to naively using the synthetic data.

5. CONCLUSIONS

We presented Synt++, a new algorithm to train speech recognition models using synthetic data. We applied Synt++ to ASR and keyword detection tasks and obtained significant improvements over training with real-only and real+synthetic datasets.

Acknowledgement: We are grateful to our colleagues Russ Webb and Barry Theobald for their valuable inputs.

6. REFERENCES

[1] Shuang Ma, Daniel Mcduff, and Yale Song, "A generative adversarial network for style modeling in a text-to-speech system," in *ICLR*, 2019.

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

[3] Ting-Yao Hu, Ashish Shrivastava, Oncel Tuzel, and Chandra Dhir, "Unsupervised style and content separation by minimizing mutual information for speech synthesis," in *Proc. ICASSP*, 2020.

[4] Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ-Skerry Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Ye Jia, Fei Ren, and Rif A. Saurous, "Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis," in *Proc. ICML*, 2018.

[5] Jen-Hao Rick Chang, Ashish Shrivastava, Hema Koppula, Xiaoshuai Zhang, and Oncel Tuzel, "Style equalization: Unsupervised learning of controllable generative sequence models," *arXiv preprint arXiv:2110.02891*, 2021.

[6] Christopher M. Bishop, *Pattern recognition and machine learning, 5th Edition*, Information science and statistics. Springer, 2007.

[7] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena, "Discriminator rejection sampling," in *Proc. ICLR*, 2019.

[8] Aditya Grover, Ramki Gummadi, Miguel Lazaro-Gredilla, Dale Schuurmans, and Stefano Ermon, "Variational rejection sampling," in *Proc. AISTATS*, 2018.

[9] 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.

[10] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," in *Proc. ICLR*, 2021.

[11] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller, "Deep voice 3: Scaling text-to-speech with convolutional sequence learning," in *Proc. ICLR*, 2018.

[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] Murali Karthick Baskar, Shinji Watanabe, Ramon Astudillo, Takaaki Hori, Lukáš Burget, and Jan Černockỳ, "Semi-supervised sequence-to-sequence asr using unpaired speech and text," *arXiv preprint arXiv:1905.01152*, 2019.

[14] Takaaki Hori, Ramon Astudillo, Tomoki Hayashi, Yu Zhang, Shinji Watanabe, and Jonathan Le Roux, "Cycle-consistency training for end-to-end speech recognition," in *Proc. ICASSP*, 2019.

[15] 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 *IEEE Spoken Language Technology Workshop (SLT)*, 2018.

[16] 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.

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

[18] Chenpeng Du and Kai Yu, "Speaker augmentation for low resource speech recognition," in *Proc. ICASSP*, 2020.

[19] Andrew Rosenberg, Yu Zhang, Bhuvana Ramabhadran, Ye Jia, Pedro Moreno, Yonghui Wu, and Zelin Wu, "Speech recognition with augmented synthesized speech," in *ASRU*, 2019.

[20] Masato Mimura, Sei Ueno, Hirofumi Inaguma, Shinsuke Sakai, and Tatsuya Kawahara, "Leveraging sequence-to-sequence speech synthesis for enhancing acoustic-to-word speech recognition," in *IEEE Spoken Language Technology Workshop (SLT)*, 2018.

[21] Zhehuai Chen, Andrew Rosenberg, Yu Zhang, Gary Wang, Bhuvana Ramabhadran, and Pedro J Moreno, "Improving speech recognition using gan-based speech synthesis and contrastive unspoken text selection.," in *Proc. Interspeech*, 2020.

[22] 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.

[23] Andrew Werchniak, Roberto Barra Chicote, Yuriy Mishchenko, Jasha Droppo, Jeff Condal, Peng Liu, and Anish Shah, "Exploring the application of synthetic audio in training keyword spotters," in *Proc. ICASSP*, 2021.

[24] James Lin, Kevin Kilgour, Dominik Roblek, and Matthew Sharifi, "Training keyword spotters with limited and synthesized speech data," in *Proc. ICASSP*, 2020.

[25] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proc. CVPR*, 2017.

[26] Antreas Antoniou, Amos Storkey, and Harrison Edwards, "Data augmentation generative adversarial networks," *arXiv preprint arXiv:1711.04340*, 2017.

[27] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L. Yuille, and Quoc V. Le, "Adversarial examples improve image recognition," in *Proc. CVPR*, 2020.

[28] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Proc. NIPS*, 2014.

[29] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015.

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

[31] 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," *arXiv preprint arXiv:1904.02882*, 2019.

[32] 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.

[33] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[34] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Peng Shi, "Neural speech synthesis with transformer network," in *AAAI*, 2019.

[35] 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," *Proc. Interspeech*, 2019.

[36] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," *ArXiv e-prints*, 2018.

[37] Ashish Shrivastava, Arnav Kundu, Chandra Dhir, Devang Naik, and Oncel Tuzel, "Optimize what matters: Training dnn-hmm keyword spotting model using end metric," in *Proc. ICASSP*, 2021.