Semi-Supervision in ASR: Sequential MixMatch and Factorized TTS-Based Augmentation

Zhehuai Chen, Andrew Rosenberg, Yu Zhang, Heiga Zen, Mohammadreza Ghodsi, Yinghui Huang, Jesse Emond, Gary Wang, Bhuvana Ramabhadran, Pedro Moreno

Google, Inc.

{zhehuai,rosenberg,ngyuzh,heigazen,ghodsi,huangyinghui,emond,wgary,bhuv,pedro}@google.com

Abstract

Semi and self-supervised training techniques have the potential to improve performance of speech recognition systems without additional transcribed speech data. In this work, we demonstrate the efficacy of two approaches to semi-supervision for automated speech recognition. The two approaches leverage vast amounts of available unspoken text and untranscribed audio. First, we present *factorized multilingual speech synthesis* to improve data augmentation on unspoken text. Next, we propose the *Sequential MixMatch* algorithm with *iterative learning* to learn from untranscribed speech. The algorithm is built on top of our online implementation of Noisy Student Training. We demonstrate the compatibility of these techniques yielding an overall relative reduction of word error rate of up to 14.4% on the voice search tasks on 4 Indic languages.

Index Terms: Speech Recognition, Speech Synthesis, Data Augmentation, Noisy Student, MixMatch

1. Introduction

Semi and self-supervision allow for *automated speech recognition* (ASR) systems to be trained without additional transcribed speech. Unspoken text can be challenging to incorporate into state of the art, end-to-end speech recognition systems (Section 2). This can be attributed to a number of factors including inconsistent text normalization arising from different sources of text. Over the last few years, text-to-speech (TTS) synthesis-based data augmentation has been shown to be a useful way to train end-to-end models using unspoken text [1–4]. We build on this work and investigate different speech synthesis training approaches with the goal of improving ASR performance. In addition, we propose to use factorized multilingual TTS which further reduces the need for high quality speech normally required to train TTS but not available for many languages.

The traditional method of training on untranscribed speech uses predictions from a model trained on supervised data to generate pseudo labels for the untranscribed data [5, 6]. These pseudo-labeled utterances are then added to the supervised data and training proceeds in an iterative fashion. Noisy Student Training (NST) [7] proposes to augment the student training data with label preserving transformations, such as SpecAugment [8]. This augmentation component is similar to FixMatch [9], where the teacher model is "weakly" augmented, and the student is augmented with a "strong" augmentation. Our work is built on top of an online version of NST, which requires no additional storage, as the pseudo labels are both generated and used for training in memory (Section 4.3). Moreover, we introduce consistency regularization on the data augmentation by the proposed *Sequential MixMatch*. The proposed method is motivated by MixMatch [10] in image classification, a semi-supervised learning method to combine data augmentation and consistent predictions. The two key components in that method, prediction combination and MixUp, are designed for classification tasks. To integrate a similar idea in speech recognition, a sequence labeling task, we redesign the algorithm in Section 4.2.

In this paper, we present a framework that leverages both unspoken text and untranscribed speech to improve ASR performance on four Indic languages. The contributions include:

1. *Sequential MixMatch* algorithm to improve semi-supervised training from untranscribed speech.
2. *Factorized multilingual TTS* to disentangle speaker, phoneme, prosody and language representations for efficient transfer learning
3. On-the-fly, in-memory implementation of NST.
4. Additive improvements (up to 14.4% relative) in ASR performance from both unpaired text and speech.

We show improvements in ASR across all investigated languages, namely, Kannada (kn), Tamil (ta), Telugu (te) and Bengali (bn) through both of these approaches with the aforementioned contributions. Moreover, we find these improvements to be complementary.

2. Related Work

Semi-supervised and unsupervised training methods are used extensively in ASR. Successful incorporation of unspoken text was demonstrated in a variety of tasks by connecting ASR and TTS via an end-to-end differentiable loss [1, 2, 4, 11]. It has also been shown that prosodically diverse synthesized speech effectively improves ASR performance [12]. Leveraging vast amounts of unpaired text through learned text representations have been explored using shared encoder representations in [13–15]. These approaches have shown great promise in ASR when combined with both transcribed speech (paired) and untranscribed speech (unpaired) [3, 16]. The most popular framework for this combination is the Deep Chain framework [1], with adversarial [17] and cycle consistency training objectives [18] providing increased robustness for ASR.

In [19], TTS was used to augment the training data with code switched material to improve overall ASR performance on Hindi-English speech. Synthesis of out-of-vocabulary terms to improve performance of rare words was exploited for Kannada ASR in [20]. A multilingual TTS model, allowing knowledge transfer across languages for training TTS systems, was introduced in [21]. However, its use to improve ASR for low resource languages has not been explored prior to this work.

More recently, self-supervision has gained considerable attention in ASR to address the lack of data in low resource languages [22,23]. Prediction combination methods [24], Noisy Student Training (NST) [7, 25] and augmentation methods

such as FixMatch [9, 26] have continued to improve unsupervised and semi-supervsed learning for ASR. MixMatch [10], RealMix [27] and ReMixMatch [28] are image classification algorithms that combine the benefits of all the data augmentation, consistency regularization and prediction combination. In this paper, we novelly extend the MixMatch framework from classification tasks to sequence labeling tasks such as ASR.

3. Factorized Multilingual TTS for ASR

Augmenting training data with hypotheses derived from speech synthesis has two major benefits. First, synthesis can be used to generate acoustic realizations of out-of-vocabulary (oov) tokens and unseen sequences. We refer to this as improving the *lexical diversity* of the training data. Second, synthesis with sufficient control and variability can generate diverse realizations of utterances, via prosodic (suprasegmental) changes or spectral modification. The control of these parameters are dependent on the TTS model itself. We refer to this as increasing *acoustic diversity* of the training data.

3.1. TTS Model

We base our TTS model on Tacotron 2D [29], which takes text sequences as input, conditioned on speaker or utterance embeddings and outputs a sequence of mel spectrogram frames. The decoder network takes the phoneme encoding from the encoder and speaker embedding from a separately trained speaker encoder [29]. We **never** synthesize audio waveforms, thereby eliminating the need for any vocoder. We directly generate mel-filter bank features as input for training ASR models. To model the prosody and increase its variability during inference, we further augment the model with a variational auto encoder (VAE) as in [30] and modify its global VAE to a hierarchical version [2]. The motivation behind this is to capture the local and global speaking styles separately and make TTS more stable. The hierarchical VAE includes a local encoder which encodes fixed two-second chunks with a one-second overlap and a global encoder encodes the whole utterance.

3.2. Factorized Multilingual TTS

TTS augmentation for ASR requires a TTS model for each language. It is challenging to scale this training across many languages for two main reasons. First, collecting high quality studio data with different speakers in many of these languages is challenging and second the use of one TTS model per language imposes large memory requirements in a joint training framework. To alleviate these issues, we train one multilingual TTS [21] model to synthesize speech in multiple languages. Subsequently, we can use this to improve the performance of multiple ASR models each recognizing speech in one language.

As shown in previous work [2, 15], speaker diversity is crucial to reap benefits from TTS-based data augmentation. Nevertheless, there are only a limited number of available speakers with formal, constrained speaking styles in TTS corpora for most languages [31]. The multilingual TTS framework can transfer speaker knowledge across languages, for example, from English to Indic languages with fewer speakers in TTS training corpora.

Most Indic languages have very few professional speakers for training TTS systems. In this work we have 2 speakers for Kannada, Tamil and Telugu and 5 for Bengali. In contrast, for US English, we have nearly three thousand speakers obtained from a variety of speaking styles and recording conditions available for TTS training (including e.g. [32]).

Text from each language is transliterated [33] to its native script and used as input to the multilingual TTS model. The phoneme tokenizer of each language is used to tokenize the native script to a phoneme sequence. All languages share a global, SAMPA-derived, phoneme set. To better control the multilingual TTS data augmentation, we propose to disentangle the following factors in the TTS model via adversarial classifiers and conditional speaker, phoneme, language, and prosody embeddings (cf. Figure 1).

- Speaker/Phoneme Factorization: We add adversarial loss of speaker classification on encoder outputs to remove the speaker information in the phoneme embedding [21]. To alleviate the instability in traditional adversarial training, we classify the top n high-resource TTS speakers separately and merge the remaining low-resource TTS speakers into a single class. In addition, we apply gradient clipping applied to speaker adversarial loss in the TTS model.
- Speaker/Language Factorization: We train language embeddings jointly with the TTS model. Language embeddings are added as input to the TTS encoder to improve phoneme embedding extraction, while augmenting the input to the decoder provides control over the accent of the TTS audio.
- Speaker/Prosody Factorization: We use a hierarchical VAE encoder [2] to represent prosody. To disentangle prosody from speaker information, we include adversarial loss of TTS speaker classification on the VAE encoder outputs. To discourage the synthesis of accented speech, we reduce the weight of KL-loss term in VAE-training.

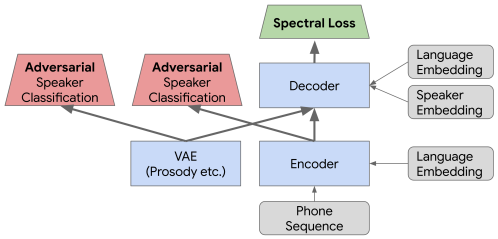

Figure 1: *Factorized multilingual TTS architecture.*

Data synthesized from multilingual TTS is augmented with transcribed speech for ASR training in a fashion similar to how monolingual synthesized data is used [2].

4. Sequential MixMatch for Semi-supervised Learning

Semi-supervised training to leverage untranscribed speech is a technique that has been used over several decades e.g. [5]. The algorithm used in this paper follows a well-proven approach: A base ASR model is trained on the available transcribed speech and used to generate hypotheses for the untranscribed speech data. Heuristic measures using confidence measures or ASR WER are used to select utterances from the untranscribed set to augment the ASR training data. This process is repeated iteratively. Central to this technique is the application of data augmentation to perturb the unlabeled data when training the next iteration of the ASR model (student model). In this work, we introduce consistency regularization on the data augmentation used in the proposed *Sequential MixMatch*. The method can be combined with *iterative training* to get additional benefits.

4.1. MixMatch in Image Processing

The proposed method is motivated by MixMatch [10] in image classification, a semi-supervised learning method to combine

data augmentation and consistent predictions. MixMatch generates pseudo labels from a combination of predictions from diverse augmented examples. The resultant label precision can be improved by introducing consistency between multiple predictions. The data and its pseudo labels are used to generate new data pairs for model training using the MixUp [34] algorithm.

4.2. Sequential MixMatch

The two key components of MixMatch, prediction combination and MixUp, were designed for static classification tasks. To integrate a similar idea in a sequence labeling task such as ASR, we redesign the algorithm as shown in Figure 2.

Prediction combination helps ensure that a classifier outputs the same class distribution for an unlabeled example even after it has been augmented, we first generate N augmented versions of a clean example, where the k-th version is denoted as $\mathbf{x}^{(k)}$ and the corresponding pseudo labels generated by the teacher model is $\mathbf{y}^{(k)}$. We introduce consistency regularization by obtaining the average prediction $\bar{\mathbf{q}}$ from the N predicted distributions as final labels for student training.

$$\bar{\mathbf{q}} = \frac{1}{N}\sum_{k=1}^{N} \mathbf{q}_{\mathbf{x}^{(k)}} = \frac{1}{N}\sum_{k=1}^{N} p(\mathbf{y}|\mathbf{x}^{(k)}) \tag{1}$$

where $\mathbf{q}_{\mathbf{x}^{(k)}}$ is the predicted distributions from $\mathbf{x}^{(k)}$, one of the N augmentations, which is modeled by the end-to-end ASR model $p(\mathbf{y}|\mathbf{x}^{(k)})$.

Notably, $\bar{\mathbf{q}}$ and $\mathbf{q}_{\mathbf{x}^{(k)}}$ are both a sequence of distributions. Hence the average is not viable if the N distributions $\mathbf{q}_{\mathbf{x}^{(k)}}$ are not time-aligned and label-aligned with each other. We firstly propose to generate the above distributions from a LAS [35] model, whose attention mechanism enables time-aligned label predictions. A second requirement is to make each prediction at the same label index l (where $l \in [1, L]$) given the same label history, denoted by $\mathbf{y}_{<l}$. We propose to condition on a common prefix from the beam search result of the teacher LAS model. We can thus calculate the average predicted distribution of each output label. We follow [10] to sharpen the average distribution in order to encourage entropy minimization [36]. We sample one label $\tilde{y}_l$ from each sharpened distribution to get the final label $\tilde{\mathbf{y}}$. Thus, $\tilde{\mathbf{y}}$ generated by a LAS teacher can be used by either RNN-T [37] or LAS student training.

$$\tilde{y}_l = \text{Sample}\big(\text{Sharpen}(\bar{q}_l)\big) \tag{2}$$

$$= \text{Sample}\left(\text{Sharpen}\left(\frac{1}{N}\sum_{k=1}^{N} p(y_l|\mathbf{x}^{(k)}, \mathbf{y}_{<l})\right)\right) \tag{3}$$

To extend MixUp to sequence labeling, while avoiding the requirement of aligning features and targets, one can replace *label MixUp* by *loss MixUp* [38]. Our preliminary experiments did not show the RNN-T loss in MixUp to be effective. Thus, rather than using a MixUp variant, we use pseudo labels from the above procedure as targets to a different but strongly augmented version of the same data for student training, similar to FixMatch [9]. Overall, we use $(N+1)$ different augmentations in generating the final data. These designs form the Sequential MixMatch algorithm.

4.3. Online Iterative Training

We integrate iterative training [7, 25] with the proposed Sequential MixMatch. We begin by iteratively training a LAS student model using a LAS teacher model. Next, we use the best LAS student model as the teacher to train a RNN-T student to convergence (referred to as one iteration).

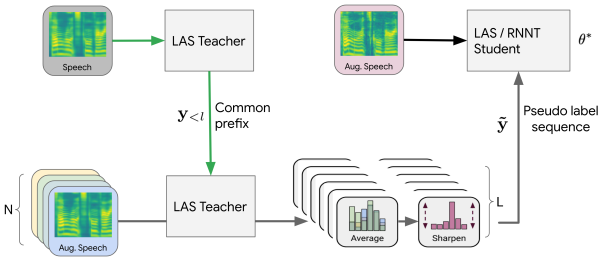

Figure 2: *Sequential MixMatch framework.*

Our approach has an additional improvement in that we run the full algorithm in memory, without storing intermediate hypotheses to disk (*Online Noisy Student Training*). In order to do this, we perform a filtering process at a per-batch level (our batch size is 4096). Within each training batch, we include a portion of labeled and unlabeled speech. On the unlabeled portion, we generate hypotheses using above Sequential MixMatch, filter out a portion of low confidence hypotheses, and then use the filtered hypotheses as pseudo labels for training along with the labeled data.

Previous iterative training uses a fixed model for generating pseudo labels of the full set of untranscribed speech. In this "online" version, we keep both the teacher and student in memory, and update the student against a predetermined schedule. At one extreme, we can update the teacher after every update, essentially using the same model as teacher and student. However, we find that this has a tendency to reinforce errors quickly. Instead, we update the teacher more slowly, after a certain number of batches. For Kannada which is the most data-scarce language among the four Indic languages, the teacher update schedule was empirically determined to be as follows: We sample 6 checkpoints randomly after 10k updates, evaluate their performance on the development test set, and update the trainer to the student with the best WER on a held out data set. We find that it is sufficient to update the teacher two times (iterations) before model convergence.

5. Experiment Setup

Experiments are conducted on 4 Indic languages. The ASR training data for all languages are anonymized and hand-transcribed, and are representative of Google's voice search traffic. The training set is created by artificially corrupting clean utterances using a room simulator and addition of various noise styles [39]. The development set is a small fraction of the training set held out for validation. The test set also contains anonymous transcribed utterances from the voice-search task. The unspoken text material for TTS augmentation includes hundred millions of anonymized websearch queries in each target language. It's larger than the transcribed speech corpus by several orders of magnitude, which makes speech synthesis of all the text data impractical. To improve efficiency of learning from unspoken text, we use contrastive language model-based data selection [15] and only keep 30 million queries for each language. The untranscribed speech data used by Sequential MixMatch is roughly 2 times that of the supervised data in each language. As this data is without human transcriptions, we cannot report WER on the training set as done in [7].

The multilingual TTS model uses a Tacotron2 TTS architecture described in [4] with hierarchical VAE [30]. The decoder is followed by a PostNet with five convolutional layers

of 512 filters with shape 5 x 1. The training data used for TTS models uses two professional speakers per Indic language. For English, we combine 80 professional TTS speakers with the speakers in the publicly available LibriTTS corpus [32].

All ASR student models use the RNN-T architecture [37, 39] It consists of a 6-layer (2048-dim LSTM followed by a 640-dim linear projection layer) encoder and a 2-layer (2048-dim RNN followed by a 640-dim projection layer) decoder with a single layer joint network. The input features consist of three stacked 80-dim log-mel features at a 10ms frame-rate, leading to a downsampled 30ms frame rate. The network is trained with a 4096 sub-word unit, i.e., word-piece vocabulary derived as described in [40]. The LAS model shares the same encoder architecture as the RNN-T student model. The decoder network is a 2-layer LSTM with 1024 hidden units per layer [35]. We use two SpecAugment [8] augmentations to generate predictions for Sequential MixMatch ($N = 2$). We design the augmentations to use a weak SpecAugment with one frequency mask and a strong SpecAugment with two time and frequency masks. All models were trained in Lingvo [41] on Tensor Processing Unit [42] slices with a batch size of 4096. The WER metric we report is transliteration-optimized WER described in [33] to accommodate mixed writing scripts frequently seen in Indic languages.

6. Results and Discussion

6.1. Factorized Multilingual TTS Augmentation

We begin this section with the results from the TTS-based augmentation approach from Section 3. The overall impact of TTS augmentation is shown in Table 1. We find the incorporation

	kn	ta	te	bn
Baseline RNN-T	29.8	20.7	25.2	15.4
+TTS'd text	27.5	19.5	24.2	14.8

Table 1: *WER Improvement from synthesized unspoken text.*

of synthesized unspoken text consistently improves ASR performance across all Indic languages, Kannada (kn), Tamil (ta), Telugu (te), Bengali (bn). The improvement ranges between 7.7% and 3.9% relative with the largest improvement seen in the most data-scarce language, Kannada.

To show the benefit of the proposed factorized multilingual TTS for ASR, we investigate the effect of augmentation using single and multilingual TTS models on Kannada in Table 2. Our first observation is that multilingual TTS, trained

Systems	Inference speakers	WER
Baseline RNN-T	-	29.8
+ kn/in TTS	kn/in	28.7
+ multilingual TTS	kn/in	28.0
+ multilingual TTS	kn/in + top en/us	28.4
+ factorized	kn/in + top en/us	27.9
+ factorized	kn/in + all en/us	27.5

Table 2: *Ablation study of multilingual TTS for ASR.*

with other Indic and American English material provides better ASR training data, even when inference is performed only on Kannada speakers (Rows 2 and 3 of Table 2). The addition of speakers from other Indic languages and en/us increases the speaker diversity of the synthesized data which results in WER improvement in Row 3 and 4. Second, we find that factorization proposed in Section 3.2 can successfully improve the multilingual TTS system which in turn helps ASR in the last two rows. The factorized model can better control the speaker, phoneme and prosody in the synthesized speech, which enables the incorporation of large amount of foreign speakers. This not only increases the WER reductions, but also supports the design of one multilingual TTS to improve multiple ASR models.

6.2. Sequential MixMatch for Semi-supervised training

Table 3 presents results on untranscribed speech using Sequential MixMatch described in Section 4 across different languages. It can be seen that the improvements from untran-

	kn	ta	te	bn
Baseline RNN-T	29.8	20.7	25.2	15.4
+ Untranscribed speech	27.7	19.5	23.9	14.6

Table 3: *WER Improvement from untranscribed speech.*

scribed speech are similar to those obtained from TTS-based data augmentation from unspoken text, ranging between 5.3% and 7.0% relative. We conduct an ablation study in Table 4 on the proposed Sequential MixMatch algorithm, by comparing it to the state-of-the-art Noisy Student Training (NST) at different steps. Results from baseline online NST described in Section 4.3 are presented in Row 1. This model is used as the teacher model in online NST in Row 2 and is replaced with a LAS in Row 3. As described in Section 4.3, we first do iterative training on the LAS, followed by one iteration of the LAS model as a teacher to train the RNN-T student. The proposed Sequential MixMatch in the last row outperforms all above systems and improves the most from untranscribed speech.

	Teacher	WER
Baseline RNN-T	-	29.8
+ Online NST	RNN-T	28.2
+ Online NST	LAS	28.0
+ Seq. MixMatch	LAS	27.7

Table 4: *Ablation study of Sequential MixMatch.*

In order to assess if these two approaches to semi-supervision are complementary, we combine the use of unspoken-text and untranscribed speech in training one ASR model. Table 5 shows that the addition of synthesized data reduces WER by 3.9–7.7% relative, and the combination with Sequential MixMatch provides an additional reduction in WER resulting in an overall reduction in WER of 5.3–14.4% relative.

	kn	ta	te	bn
Baseline RNN-T	29.8	20.7	25.2	15.4
+ Unspoken text	27.5	19.5	24.2	14.8
+ Untranscribed speech	25.5	19.3	23.9	14.6

Table 5: *Overall results of semi-supervision from unspoken text and untranscribed speech*

7. Conclusion

We have demonstrated that using speech synthesis to train speech recognition on unspoken text is effective on four Indic languages. In describing this effort, we have presented the *factorized multilingual speech synthesis* training and inference to improve data augmentation of unspoken text for ASR. We have also shown that *Sequential MixMatch* algorithm is a better semi-supervised training technique on untranscribed speech. Finally we demonstrated that these two approaches are highly complementary resulting in an overall 14.4% WER reduction.

8. References

[1] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while speaking: Speech chain by deep learning," in *2017 ASRU*. IEEE, 2017, pp. 301–308.

[2] A. Rosenberg, Y. Zhang, B. Ramabhadran, Y. Jia, P. Moreno, Y. Wu, and Z. Wu, "Speech recognition with augmented synthesized speech," in *2019 ASRU*. IEEE, 2019, pp. 996–1002.

[3] S. Karita *et al.*, "Semi-supervised end-to-end speech recognition using text-to-speech and autoencoders," in *ICASSP*. IEEE, 2019.

[4] G. Wang, A. Rosenberg, Z. Chen, Y. Zhang, B. Ramabhadran, Y. Wu, and P. Moreno, "Improving speech recognition using consistent predictions on synthesized speech," in *ICASSP*. IEEE, 2020, pp. 7029–7033.

[5] T. Kemp and A. Waibel, "Unsupervised training of a speech recognizer: Recent experiments," in *Sixth European Conference on Speech Communication and Technology*, 1999.

[6] L. Lamel, J.-L. Gauvain, and G. Adda, "Lightly supervised and unsupervised acoustic model training," *Computer Speech & Language*, vol. 16, no. 1, pp. 115–129, 2002.

[7] D. S. Park, Y. Zhang, Y. Jia, W. Han, C.-C. Chiu, B. Li, Y. Wu, and Q. V. Le, "Improved noisy student training for automatic speech recognition," *arXiv preprint arXiv:2005.09629*, 2020.

[8] 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," *Proc. Interspeech 2019*, pp. 2613–2617, 2019.

[9] e. a. Sohn, Kihyuk, "FixMatch: Simplifying semi-supervised learning with consistency and confidence," *arXiv preprint arXiv:2001.07685*, 2020.

[10] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," *arXiv preprint arXiv:1905.02249*, 2019.

[11] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. Le Roux, "Cycle-consistency training for end-to-end speech recognition," in *ICASSP 2019*. IEEE, 2019, pp. 6271–6275.

[12] G. Sun *et al.*, "Generating diverse and natural text-to-speech samples using a quantized fine-grained VAE and autoregressive prosody prior," in *ICASSP*. IEEE, 2020.

[13] D. He, Y. Xia, T. Qin, L. Wang, N. Yu, T.-Y. Liu, and W.-Y. Ma, "Dual learning for machine translation," in *Advances in neural information processing systems*, 2016.

[14] T. Hayashi, S. Watanabe, Y. Zhang, T. Toda, T. Hori, R. Astudillo, and K. Takeda, "Back-translation-style data augmentation for end-to-end ASR," in *SLT*. IEEE, 2018.

[15] Z. Chen, A. Rosenberg, Y. Zhang, G. Wang, B. Ramabhadran, and P. Moreno, "Improving speech recognition using GAN-based speech synthesis and contrastive unspoken text selection," in *Interspeech*, 2020.

[16] Y. Ren, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Almost unsupervised text to speech and automatic speech recognition," *arXiv preprint arXiv:1905.06791*, 2019.

[17] A. H. Liu, H.-y. Lee, and L.-s. Lee, "Adversarial training of end-to-end speech recognition using a criticizing language model," in *ICASSP*. IEEE, 2019, pp. 6176–6180.

[18] M. K. Baskar *et al.*, "Semi-supervised sequence-to-sequence ASR using unpaired speech and text," *arXiv preprint arXiv:1905.01152*, 2019.

[19] Y. Sharma, B. Abraham, K. Taneja, and P. Jyothi, "Improving low resource code-switched ASR using augmented code-switched TTS," 2020.

[20] S. Murthy, D. Sitaram, and S. Sitaram, "Effect of TTS generated audio on OOV detection and word error rate in ASR for low-resource languages," in *Interspeech*, 2018.

[21] Y. Zhang *et al.*, "Learning to speak fluently in a foreign language: Multilingual speech synthesis and cross-language voice cloning," *arXiv preprint arXiv:1907.04448*, 2019.

[22] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *arXiv preprint arXiv:2006.11477*, 2020.

[23] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020.

[24] X. Cui, J. Huang, and J.-T. Chien, "Multi-view and multi-objective semi-supervised learning for hmm-based automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 1923–1935, 2012.

[25] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves ImageNet classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 687–10 698.

[26] F. Weninger, F. Mana, R. Gemello, J. Andrés-Ferrer, and P. Zhan, "Semi-supervised learning with data augmentation for end-to-end ASR," *arXiv preprint arXiv:2007.13876*, 2020.

[27] V. Nair, J. F. Alonso, and T. Beltramelli, "Realmix: Towards realistic semi-supervised deep learning algorithms," *arXiv preprint arXiv:1912.08766*, 2019.

[28] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel, "ReMixMatch: Semi-supervised learning with distribution alignment and augmentation anchoring," *arXiv preprint arXiv:1911.09785*, 2019.

[29] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren *et al.*, "Transfer learning from speaker verification to multispeaker text-to-speech synthesis," in *Advances in neural information processing systems*, 2018, pp. 4480–4490.

[30] W.-N. Hsu *et al.*, "Hierarchical generative modeling for controllable speech synthesis," *arXiv preprint arXiv:1810.07217*, 2018.

[31] A. Prakash and H. A. Murthy, "Generic indic text-to-speech synthesisers with rapid adaptation in an end-to-end framework," *arXiv preprint arXiv:2006.06971*, 2020.

[32] H. Zen *et al.*, "LibriTTS: A corpus derived from librispeech for text-to-speech," *arXiv preprint arXiv:1904.02882*, 2019.

[33] J. Emond, B. Ramabhadran, B. Roark, P. Moreno, and M. Ma, "Transliteration based approaches to improve code-switched speech recognition performance," in *2018 SLT*. IEEE, 2018, pp. 448–455.

[34] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "MixUp: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[35] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[36] Y. Grandvalet, Y. Bengio *et al.*, "Semi-supervised learning by entropy minimization." in *CAP*, 2005, pp. 281–296.

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

[38] L. Meng, J. Xu, X. Tan, J. Wang, T. Qin, and B. Xu, "MixSpeech: Data augmentation for low-resource automatic speech recognition," *arXiv preprint arXiv:2102.12664*, 2021.

[39] Y. He *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *ICASSP*. IEEE, 2019, pp. 6381–6385.

[40] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[41] J. Shen *et al.*, "Lingvo: a modular and scalable framework for sequence-to-sequence modeling," *arXiv preprint arXiv:1902.08295*, 2019.

[42] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, 2017.