INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Semi-supervised ASR by End-to-end Self-training

Yang Chen[1], *Weiran Wang*[2], *Chao Wang*[3]

[1]Georgia Institute of Technology, USA
[2]Salesforce Research, USA
[3]Amazon Alexa, USA

yang.chen@cc.gatech.edu, weiran.wang@salesforce.com, wngcha@amazon.com

Abstract

While deep learning based end-to-end automatic speech recognition (ASR) systems have greatly simplified modeling pipelines, they suffer from the data sparsity issue. In this work, we propose a self-training method with an end-to-end system for semi-supervised ASR. Starting from a Connectionist Temporal Classification (CTC) system trained on the supervised data, we iteratively generate pseudo-labels on a mini-batch of unsupervised utterances with the current model, and use the pseudo-labels to augment the supervised data for immediate model update. Our method retains the simplicity of end-to-end ASR systems, and can be seen as performing alternating optimization over a well-defined learning objective. We also perform empirical investigations of our method, regarding the effect of data augmentation, decoding beamsize for pseudo-label generation, and freshness of pseudo-labels. On a commonly used semi-supervised ASR setting with the Wall Street Journal (WSJ) corpus, our method gives 14.4% relative WER improvement over a carefully-trained base system with data augmentation, reducing the performance gap between the base system and the oracle system by 46%.

Index Terms: Semi-supervised ASR, Self-training, CTC

1. Introduction

One challenge faced by modern ASR systems is that, with ever enlarged model capacity, large amount of labeled data are required to thoroughly train them. Unfortunately, collecting and transcribing huge dataset is expensive and time-consuming. As a result, semi-supervised ASR has been an important research direction, with the goal of leveraging a large amount of unlabeled data and a much smaller amount of labeled data for training. One of the simplest methods in this setting is self-training, which uses the decoding results or pseudo-labels on unsupervised data, often at the word level, to augment supervised training. It has been shown to be very effective with traditional ASR pipelines [1, 2, 3, 4].

In this work, we propose a novel framework for self-training in an end-to-end fashion. Starting from a carefully-trained Connectionist Temporal Classification (CTC, [5]) system, we alternate the following two procedures: generating pseudo-labels using a token-level decoder on a mini-batch of unsupervised utterances, and augmenting the just decoded (input, pseudo-label) pairs for supervised training. We show that this method can be derived from alternating optimization of a unified objective, over the acoustic model and the non-observed labels of unsupervised data. The two procedures effectively reinforce each other, leading to increasingly accurate models.

We emphasize a few important aspects of our method, which distinguish our work from others (detailed discussions on related work are provided later):

- The pseudo-labels we use are discrete, token-level label sequences, rather than per-frame soft probabilities.
- The pseudo-labels are generated on the fly, rather than in one shot, since fresh labels are of higher quality than those produced from a stale model.
- We perform data augmentation not only on supervised data, but also on unsupervised data.

These modeling choices, which lead to performance gain over alterantives, are backed up by empirical results. We demonstrate our method on the WSJ corpus ([6], LDC catalog numbers LDC93S6B and LDC94S13B). Our method improves PER by 31.6% relative on the development set, and WER by 14.4% relative on the test set from a well-tuned base system, bridging 46% of the gap between the base system and the oracle system trained with ground truth labels of all data.

In the rest of this paper, we review the supervised component of our method in Sec. 2, give detailed description of the proposed method in Sec. 3, compare with related work for semi-supervised ASR in Sec. 4, provide comprehensive experimental results in Sec. 5, and conclude with future directions in Sec. 6.

2. Supervised learning for ASR

Before describing the proposed method, we review the supervised component in our system—CTC with data augmentation.

2.1. End-to-end ASR with CTC

Given an input sequence $X = (\mathbf{x}_1, ..., \mathbf{x}_T)$ and the corresponding label sequence $Y = (\mathbf{y}_1, ..., \mathbf{y}_L)$, CTC introduces an additional <blank> token and defines the conditional probability

$$\mathbb{P}(Y|X) = \sum_{p \in \mathcal{B}^{-1}(Y)} \prod_{j=1}^{T} \mathbb{P}(p_j|X)$$

where $\mathcal{B}^{-1}(Y)$ is the set of all paths (frame alignments) that would reduce to Y after removing repetitions and <blank> tokens, and $\mathbb{P}(p_j|X)$ is the posterior probability of token p_j at the j-th frame by the acoustic model. The underlying assumption is that conditioned on the entire input sequence X, the probability for a path p decouples over the frames. The CTC loss for one utterance (X, Y) is then defined as $\mathcal{L}_{CTC}(X, Y) = -\log \mathbb{P}(Y|X)$. CTC training minimizes the averaged loss over a set of labeld utterances. It is well known that after training, the per-frame posteriors from the acoustic model tend to be peaky, and at most frames the most probably token is <blank> with high confidence, indicating "no emission".

Due to the abovementioned independence assumption, CTC does not explicitly model transition probabilities between labels, and thus decoding—the problem of $\max_Y \mathbb{P}(Y|X)$—is relatively straightforward. The simplest decoder for CTC

http://dx.doi.org/10.21437/Interspeech.2020-1280

is the greedy one, which picks the most probably token at each frame and then collapses them by removing repetitions and <blank>'s; we will be mostly using this decoder as it is extremely efficient. One can improve the greedy decoder by maintaining a list of W hypothesis at each frame, leading to a beam search decoder with beamsize W. When modeling units are subwords but word-level hypothesis are desired, one can incorporate lexicon and language models, which can be implemented efficiently in the WFST framework [7]. We do not use word-level decoder for generating pseudo-labels since it is much slower than token-level beam search, and it depends on the availability of an in-domain language model. In this work, we only use word-level decoder for evaluating the word error rates (WERs). It should be noted that, our self-training method can make use of the attention-based systems [8, 9] as well. We use CTC mainly due to its simplicity and efficiency in decoding, for generating pseudo-labels on the fly.

2.2. Data augmentation

To alleviate the data sparsity issue, a natural approach that does not require unsupervised data is to augment the training data with distorted versions. And various data augmentation techniques have demonstrated consistent improvement for ASR [10, 11, 12, 13]. This simple way of obtaining supervised training signal helps us to improve our base system, which in turn generates pseudo-labels with higher quality.

In this work, we adopt the speed perturbation and spectral masking techniques from [13]. Both techniques perturb inputs at the spectrogram level. One can view the input utterance as an image of dimension $D \times T$ where D corresponds to the number of frequency bins, and T the number of frames. Speed perturbation performs linear interpolation along the time axis, as in an image resizing operation; two speed factors 0.9 and 1.1 are used. Spectral masking selects m_F segments of the input in the frequency axis with random locations, whose widths are drawn uniformly from $\{0, 1, \ldots, n_F\}$, and similarly select m_T segments in the time axis, with widths up to n_T. We perform grid search of hyperparameters for the supervised CTC system, and set $m_F = 1$, $n_F = 8$, $m_T = 2$, $n_T = 16$ throughout.

3. Leveraging unsupervised data with self-training

After a base system is sufficiently trained on supervised data, it can be used to predict labels on the originally non-transcribed data. If we take the confident predictions and assume that they are correct, we can add the input and the predictions (pseudo-labels) into training. If the noise in pseudo-labels is sufficiently low, the acoustic model can benefit from the additional training data to obtain improved accuracy. We propose to repeat the pseudo-label generation and augmented training steps, so as to have the two reinforce each other, and to continuously improve both. In our method, for each update, we generate pseudo-labels for a mini-batch of unsupervised utterances using the current acoustic model with beam search, and compute the CTC losses for these utterances based on their most probable hypothesis. The losses for unsupervised utterances are discounted by a factor $\gamma > 0$ to accommodate label noise, and combined with the CTC loss for supervised data to derive the next model update. A schematic diagram of our method is provided in Fig. 1.

Equivalently, we can formulate our method as minimizing

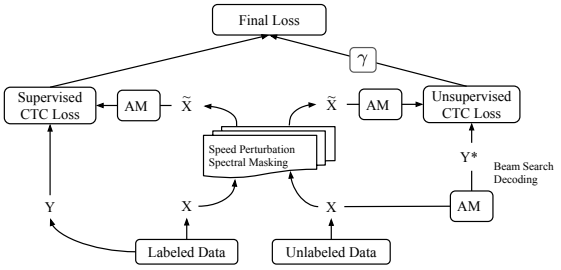

Figure 1: *Our self-training method for semi-supervised ASR.*

the following objective:

$$\min_{\theta, \{Y_j^*\}} \frac{1}{N_l} \sum_{i=1}^{N_l} \mathcal{L}(X_i, Y_i; \Theta) + \frac{\gamma}{N_u} \sum_{j=1}^{N_u} \mathcal{L}(X_j, Y_j^*; \Theta) \quad (1)$$

where $\mathcal{L}(X, Y)$ denotes the CTC loss, we have N_l supervised utterances and N_u unsupervised utterances, Θ denotes weight parameters in the acoustic model, and we also include the (non-observed) label sequences $\{Y_j^*\}$ of unsupervised utterances as variables. This is a well-defined learning objective, and our method effectively performing alternating optimization over Y_j^* (by beam search) and the weights Θ (by gradient descent) over mini-batches. Additionally, we can perform data augmentation on the unsupervised data, by using the label sequence decoded from the original data on its distorted versions. We will show experimentally that augmenting unsupervised data is as effective as augmenting supervised data.

Our method is motivated by and similar to unsupervised data augmentation (UDA, [14]) for semi-supervised learning, in that both methods use pseudo-labels and data augmentation on unsupervised data. But there is a crucial difference between the two: UDA uses soft targets (previous model output) for calculating the unsupervised loss, which encourages the model not to deviate much from that of the previous step, and in fact if there is no data augmentation, the loss on unsupervised data would be zero and has no effect for learning; in contrast, we use the discrete label sequence—output of the beam search decoder on soft targets—on each unsupervised utterance, which provides stronger supervised signals. While [14] has not worked on sequence data, we have implemented a sequence version of it, by using the per-frame posterior probabilities as soft targets, and minimizing the cross-entropy loss between soft targets and model outputs at each frame; otherwise the implementation of UDA mirrors that of our method. As demonstrated later, our method outperforms UDA by a large margin.

In view of the peaky per-frame posterior distributions from CTC models, we think our approach has the advantage that the pseudo-labels are naturally high confidence predictions, relieving us from setting a threshold for discretizing soft probabilities. Although the alignment or locations of non-<blank> tokens can be imprecise from CTC systems, it is not an issue as we only use the label sequence but not its alignment in computing the unsupervised CTC loss, which marginalizes all possible alignments. In this regard, end-to-end systems give a more elegant formulation for self-training, than traditional hybrid systems which rely on alignments.

4. Related work

Semi-supervised ASR has been studied for a long time, and self-training has been one of the most successful approaches for traditional ASR systems (see, e.g., [1, 2, 3] and references therein). It is observed that in self-training, the quality of the pseudo-labels plays a crucial role, and much of the research is dedicated to measuring the confidence of pseudo-labels and selecting high confidence ones for supervised training [2, 3]. The issue of label quality becomes even more prominent with LSTM-based acoustic models, which have high memorization capability [15]. In similar spirit, [4] have used a student-teacher learning approach on hybrid systems, to improve accuracy of student using soft targets provided by the teacher on a million hours of non-transcribed data.

Aside from self-training, cycle consistency regularization [16, 17] has been applied to semi-supervised ASR. [18, 19, 20, 21, 22] leverage unpaired speech and text data by combining ASR with Text-to-Speech (TTS) modules, with a training loss that encourages pseudo-labels from ASR to reconstruct audio features well with the TTS system, and TTS outputs to be recognized by ASR. The authors have proposed different techniques to allow gradient backpropagation through the modules, and to alleviate the audio information loss during text decoding. Alternatively, [23] maps audio data with encoder of the ASR model, and maps text with another encoder to a common space, from which text is predicted (from the ASR side) or reconstructed (from the text side) with a shared decoder; an additional regularization term is used to encourage representations of paired audio and text to be similar. The common intuition behind these work is that of auto-encoders, the most straightforward method for unsupervised learning. On the other hand, [24] use adversarial training to encourage ASR output on unsupervised data to have similar distribution as that of unpaired text data, with a criticizing language model. Our model is much simpler than the above ones, in that we do not have additional neural network models for the text modality; rather, an efficient decoder is used to discretize the acoustic model outputs, and the pseudo-labels are immediately applied to acoustic model training as targets.

Concurrent to our initial submission, the authors of [25] also adopted an end-to-end self-training approach. A few differences between our work and [25] are as follows: first, we evaluate our method with a CTC-based ASR model whereas they use an attention-based model; second, we use data augmentation on both labeled and unlabeled data and show that both are useful, whereas they do not; third, our method is simpler as we use neither word-level language model nor ensemble methods for generating pseudo-labels; finally, our pseudo-labels are generated on the fly, where they generate pseudo-labels on the entire unlabeled dataset once. More recent studies [26, 27] have similarly shown the effectiveness of data augmentation for unsupervised data in self-training.

5. Experiments

To demonstrate the effectiveness of the proposed method, we follow a commonly used semi-supervised ASR setup with the WSJ corpus [20, 21, 23]. We use the *si84* partition (7040 utterances) as the supervised data, and the *si284* partition (37.3K utterances) as unsupervised data. The *dev93* partition (503 utterances) is used as development set for all hyper-parameter tuning, and the *eval92* partition (333 utterances) as the test set. For input features, we extract 40 dimensional LFBEs with a window size of 25ms and hop size of 10ms from the audio recordings,

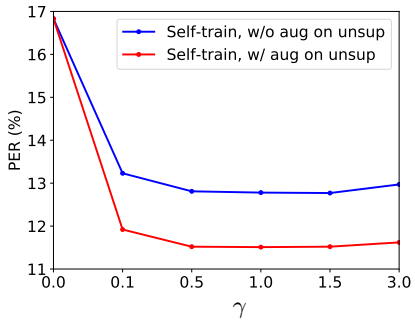

Figure 2: *Performance of our method on dev set for different* γ.

and perform per-speaker mean normalization. We stack every 3 consecutive input frames to reduce input sequence length (after data augmentation), which speeds up training and decoding.

The token set used by our CTC acoustic models are the 351 position-dependent phones together with the `<blank>` symbol, generated by the Kaldi *s5* recipe [28]. Acoustic model training is implemented with Tensorflow [29], and we use its beam search algorithm for generating pseudo-labels (with a beamsize W) and for evaluating PERs on dev/test (with a fixed beamsize of 20). To report word error rate (WER) on evaluation sets, we adopt the WFST-based framework [7] with the lexicon and the trigram language model with a 20K vocabulary size provided by the recipe, and perform beam search using Kaldi's `decode-faster` with beamsize 20. Different positional versions of the same phone are merged before word decoding, and we use the phone counts calculated from *si84* to convert posterior probability (acoustic model output) to likelihood.

Our acoustic model consists of 4 bi-directional LSTM layers [30] of 512 units in each direction. For model training, we use ADAM [31] with an initial learning rate tuned by grid search. We apply dropout [32] with rate tuned over $\{0.0, 0.1, 0.2, 0.5\}$, which consistently improves accuracy. We use the dev set PER, evaluated at the end of each training epoch, as the criterion for hyperparameter search and model selection.

5.1. Base system with data augmentation

As mentioned before, we will use a base system trained only on the supervised data to kick off semi-supervised training. For this system, we set the mini-batch size to 4 and each model is trained up to 40 epochs. We apply data augmentation as described in Sec. 2.2, which effectively yields a 3x as large supervised set due to speed perturbation. In Table 1, we give PERs of the base system and another trained without augmentation. Observe that data augmentation provides sizable gain over training on clean data only (18.52% vs. 16.83% for dev PER), leading to higher pseudo-label quality. We will always use data augmentation on supervised data from now on.

5.2. Continue with self-training

Initialized from the base system, we now continue training with our semi-supervised objective (1). Each model update is computed with 8 supervised utterances and 32 unsupervised utterances (since *si284* is about 4 times the size of *si84*, this allows us to process both supervised and unsupervised once in each epoch). The number of unsupervised utterances for each update is not a critical parameter, as the label nosie can be controlled by γ. By grid search, we set the dropout rate to 0.2, and initial learning rate to 0.0001 which is 5 times smaller than that for training the initial base model, and this has the effect of dis-

Table 1: *Performance (measured by %PER) of different methods on dev and test sets.*

Model	*dev93*	*eval92*
CTC w/o DataAug	18.52	13.54
CTC base system	16.83	11.98
Self-train W=1	**11.51**	**8.64**
w/o DataAug on unsup	12.77	
UDA	14.27	
One-shot pseudo-labels ($W = 20$)	13.68	

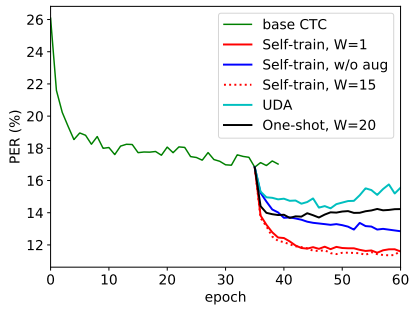

Figure 3: *Learning curves on dev set for* $\gamma = 1.0$. *Semi-supervised learning starts from* 36*-th epoch of base model.*

couraging the model to deviate too much from the base model. Each model is trained for up to another 30 epochs. We first set the beam size $W = 1$ which corresponds to the greedy decoder, for generating pseudo-labels on the fly. We train two set of models, one with data augmentation on unsupervised utterances, and the other one without; but we augment supervised utterances in both cases. The dev PERs for different values of trade-off parameter γ are given in Fig. 2, and $\gamma = 0$ corresponds to the base system. Our method performs well for a wide range of γ. The optimal γ is around 1.0 in both settings, and the performance does not degrade much with $\gamma > 1$, indicating that noise within pseudo-labels is tolerated to a large degree. Furthermore, augmenting the unsupervised data greatly improves the final accuracy.

To show that pseudo-label generation and supervised training with pseudo-labels reinforces each other, we provide in Fig. 3 the learning curve of dev PER vs. epoch for the models with $\gamma = 1.0$. The dev set accuracy improves steadily over time, with significant PER reductions in the first a few epochs from the base model.

5.3. Effect of beam size W

We now explore the effect of larger W, which intuitively shall give higher pseudo-label quality. For this experiment, we fix other hyperparameters to values found at $W = 1$. In Table 2, we give the dev PER, as well as the training time for W in $\{1, 5, 10, 15\}$. Learning curve with $W = 15$ is plotted in Fig. 3. It turns out, with larger W, we can slightly improve the final PER, at the cost of much longer training time (mostly from beam search). Therefore, we recommend using small W with a good base model.

5.4. Comparison with UDA

We now show that hard labels are more useful than soft targets, by comparing with UDA, which replaces the CTC loss on unsupervised data with cross-entropy computed with posteriors from previous model. We also use data augmentation on unsupervised data, and posteriors are interpolated in the same way as in speed perturbation for inputs. We tune the tradeoff parameter

Table 2: *Dev set performance (measured by %PER) obtained by our method with different* W*, together with training times, measured as averaged time in seconds spent by each model update, including forwarding and decoding the* 32 *unsupervised utterances and supervised training on* $8 + 32$ *utterances. Training time is recorded with a single Tesla K80.*

	$W = 1$	$W = 5$	$W = 10$	$W = 15$
dev PER (%)	11.51	11.46	11.39	11.30
Time / Update	4.72	7.88	12.10	16.53

Table 3: *Performance (measured by %WER) of previous work and our methods on* eval92*.*

Model	WER
[34] (attention, train on *si84*, unsup on *si284* by ASR+TTS)	20.30
RNN-CTC [35], train on *si84*	13.50
Our CTC, train on *si84*, w/o DataAug	13.22
Base system	11.43
UDA	10.93
One-shot pseudo-labels	10.67
Self-training, $W = 1$	**9.78**
EESEN CTC [7], train on *si284*	7.87

γ by grid search, and the best performing model (with $\gamma = 0.1$) gives a dev PER of 14.56% and learning curve in Fig. 3. The observation that hard labels outperform soft targets is in line with that of [33] for teacher-student learning with CTC.

5.5. Comparison with one-shot pseudo-labels

To further demonstrate the importance of fresh pseudo-labels, we compare with a more widely used approach where the pseudo-labels are generated once on the entire unsupervised dataset with the base model. We do so with a large decoding beam size $W = 20$, and then continue training from the base model with objective (1) without updating the pseudo-labels again. This approach does clearly improve over the base system with a dev PER of 13.68%, but not as much as our method with $W = 1$. Its learning curve is shown in Fig. 3, and the curve plateaus more quickly than those of our method.

5.6. Results summary

In Table 3 we give WERs of different methods on *eval92*. The recent work [34] which uses the same data partition for semi-supervised learning with attention models is also included. To put our results in close context, we have included the CTC model from [35] trained on *si84* only. Our method with $W = 1$ gives a relative 31.6% dev PER reduction (16.83%→11.51%), and a relative 14.4% test WER reduction (11.43%→9.78%) over a carefully-trained base system with data augmentation, effectively reducing the performance gap between the base system (11.43%) and the oracle system (7.87%) by 46%.

6. Future directions

As for future directions, we believe that word-level decoding, which incorporates lexicon and an in-domain language model, can further improve the quality of pseudo-labels after converting the word sequence back to token sequence (see, e.g., [36]), at the cost of longer decoding time. Another promising model to be used in our method is RNN-transducer [37], which has a built-in RNN LM to model label dependency and to improve token-level decoding. Furthermore, for larger W one may consider the top a few hypotheses, and use all of them for computing the loss on unsupervised data [38, 25].

7. References

[1] T. Kemp and A. Waibel, "Unsupervised training of a speech recognizer: recent experiments," in *Eurospeech*, 1999.

[2] Y. Huang, D. Yu, Y. Gong, and C. Liu, "Semi-supervised GMM and DNN acoustic model training with multi-system combination and confidence re-calibration," in *Interspeech*, 2013.

[3] S. Thomas, M. L. Seltzer, K. Church, and H. Hermansky, "Deep neural network features and semi-supervised training for low resource speech recognition," in *ICASSP*, 2013.

[4] S. H. K. Parthasarathi and N. Strom, "Lessons from building acoustic models with a million hours of speech," in *ICASSP*, 2019.

[5] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[6] D. B. Paul and J. M. Baker, "The design for the wall street journal-based CSR corpus," in *Proceedings of the workshop on Speech and Natural Language*, 1992.

[7] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *ASRU*, 2015.

[8] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *NIPS*, 2015.

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

[10] N. Jaitly and G. Hinton, "Vocal tract length perturbation (VTLP) improves speech recognition," in *ICML*, 2013.

[11] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Interspeech*, 2015.

[12] Y. Zhou, C. Xiong, and R. Socher, "Improved regularization techniques for end-to-end speech recognition," arXiv:1712.07108 [cs.CL].

[13] D. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," Apr. 18 2019, arXiv:1904.08779 [eess.AS].

[14] Q. Xie, Z. Dai, E. Hovy, M. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," in *arXiv:1904.12848*, 2019.

[15] Y. Huang, Y. Wang, and Y. Gong, "Semi-supervised training in deep learning acoustic model," in *Interspeech*, 2016.

[16] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017.

[17] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *ACL*, 2016.

[18] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while speaking: Speech chain by deep learning," in *ASRU*, 2017.

[19] ——, "Machine speech chain with one-shot speaker adaptation," in *Interspeech*, 2018.

[20] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. L. Roux, "Cycle-consistency training for end-to-end speech recognition," in *ICASSP*, 2019.

[21] M.-K. Baskar, S. Watanabe, R. Astudillo, T. Hori, L. Burget, and J. ernock, "Semi-supervised sequence-to-sequence ASR using unpaired speech and text," in *Interspeech*, 2019.

[22] Y. Ren, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T. Liu, "Almost unsupervised text to speech and automatic speech recognition," in *ICML*, 2019.

[23] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-supervised end-to-end speech recognition," in *Interspeech* 2018.

[24] A. H. Liu, H. Lee, and L. Lee, "Adversarial training of end-to-end speech recognition using a criticizing language model," in *ICASSP*, 2019.

[25] J. Kahn, A. Lee, and A. Hannun, "Self-training for end-to-end speech recognition," in *ICASSP*, 2020.

[26] Q. Xu, T. Likhomanenko, J. Kahn, A. Hannun, G. Synnaeve, and R. Collobert, "Interative pseudo-labeling for speech recognition," in *arXiv:2005.09267*, 2020.

[27] 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," in *arXiv:2005.09629*, 2020.

[28] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *ASRU*, 2011.

[29] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, and et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[31] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Int. Conf. Learning Representations*, 2015.

[32] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 2014.

[33] G. Kurata and K. Audhkhasi, "Improved knowledge distillation from bi-directional to uni-directional LSTM CTC for end-to-end speech recognition," in *SLT*, 2019.

[34] M. Baskar, S. Watanabe, R. Astudillo, T. Hori, L. Burget, and J. Černocký, "Self-supervised sequence-to-sequence ASR using unpaired speech and text," Apr. 30 2019, arXiv:1905.01152 [eess.AS].

[35] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *ICML*, 2014.

[36] Y. Chen, W. Wang, I.-F. Chen, and C. Wang, "Data techniques for online end-to-end speech recogntion," in Submission.

[37] A. Graves, "Sequence transduction with recurrent neural networks," in *ICML Workshop on Representation Learning*, 2012.

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