ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9413375

EAT: ENHANCED ASR-TTS FOR SELF-SUPERVISED SPEECH RECOGNITION

Murali Karthick Baskar[ϕ], Lukáš Burget[ϕ], Shinji Watanabe[†], Ramon Fernandez Astudillo[π], and Jan "Honza" Černocký[ϕ]

[ϕ]Brno University of Technology, [†] Johns Hopkins University, [π]IBM Research,
baskar@fit.vutbr.cz

ABSTRACT

Self-supervised ASR-TTS models suffer in out-of-domain data conditions. Here we propose an enhanced ASR-TTS (EAT) model that incorporates two main features: 1) The ASR→TTS direction is equipped with a language model reward to penalize the ASR hypotheses before forwarding it to TTS. 2) In the TTS→ASR direction, a hyper-parameter is introduced to scale the attention context from synthesized speech before sending it to ASR to handle out-of-domain data. Training strategies and the effectiveness of the EAT model are explored under out-of-domain data conditions. The results show that EAT reduces the performance gap between supervised and self-supervised training significantly by absolute 2.6% and 2.7% on Librispeech and BABEL respectively.

Index Terms— cycle-consistency, self-supervision, sequence-to-sequence, speech recognition

1. INTRODUCTION

The application of the sequence-to-sequence architecture [1] to ASR and TTS models paved way to perform self-supervised training by simple integration of ASR and TTS. Recent works on self-supervised training [2, 3, 4, 5] leveraging unpaired speech and text have shown higher performance compared to other unsupervised training approaches. Most of the research in self-supervised ASR is done in effectively integrating ASR and TTS such that it is differentiable [6] and easily trainable. However, ASR and TTS are exploited in disconnected fashion by synthesizing speech using TTS [7, 8, 9] and improving ASR through data augmentation. These techniques focus on the synthesis part and rely on text only data from unpaired sets to improve recognition performance. The work in [10] also improves ASR performance, by using a language model as a hypothesis scorer and applying self-training techniques over the resulting corrected pseudo-labels. In [11], the authors apply self-supervision through pre-training with the help of a BERT model to improve ASR performance with unpaired data. BERT has also been used as a effective pre-training technique with contrastive loss in [12] by training in self-supervised fashion.

A recent work [13] on semi-supervised sequence-to-sequence ASR has applied consistency training and has shown effectiveness with unlabeled speech data. Our previous work called ASR-TTS [4] used cycle-consistency training with REINFORCE and showed gains on standard speech datasets. However, our experiments with other corpora showed that the model suffers under the out-of-domain data condition and has further room for improvement in-terms of training and architecture.

In this work, we investigate methods to improve the robustness of the cycle-consistency approach in limited data and out-of-domain scenarios. The contributions can be itemized as follows

- We incorporate a pre-trained RNNLM regularization term in the ASR REINFORCE loss for speech only (SO) training, increasing its robustness to bad latent ASR hypotheses.
- We introduce a hyper-parameter for text-only (TO) training, to attenuate the influence of the ASR encoder by scaling the attention-encoded context. This allows us to reduce the focus on acoustic information when the latent speech quality is poor, effectively alternating between ASR and a more language-model-like behaviour.
- We incorporate latest training strategies and architectures such as data augmentation and data annealing. The TTS module is also built using the Transformer architecture, which shows higher robustness and memory efficiency. Multi-head attention is used as ASR encoder layers to attain additional gains and for reduced model complexities.
- We show that these techniques greatly improve performance and particularly attain the target goal achieving good performance in limited data and out-of-domain scenarios with cycle consistency techniques.

We call this improved model *enhanced ASR-TTS (EAT)*. Experiments are conducted on Librispeech and the BABEL-Pashto datasets and show the efficacy of our EAT model. Results are further compared with state-of-the-art (SotA) results in literature.

2. PRELIMINARIES

Our previous work, ASR-TTS [4] is a self-supervised training system built to handle both speech only (SO) and text only (TO) data using a cycle-consistency training regime. ASR-TTS training approach containing two pipelines: 1) ASR→TTS pipeline to train using SO dataset 2) TTS→ASR pipeline to train using TO dataset. The SO data x is fed to ASR→TTS pipeline to reconstruct x as $\hat{x}$. The pipeline is trained with an expected loss (approximated through REINFORCE). The text only data y is fed to the TTS→ASR pipeline to predict the text $\hat{y}$ and is trained with a cross-entropy loss objective. Both pipelines act as auto-encoders allowing to perform self-supervised training on either SO dataset $\mathcal{D}_u^s$ or TO dataset $\mathcal{D}_u^t$. The ASR architecture used in our ASR-TTS model is built using an RNN based sequence-to-sequence model and the Tacotron is used for TTS.

All the authors from Brno university of Technology are supported by Czech National Science Foundation (GACR) project "NEUREM3" No. 19-26934X and Czech Ministry of Education, Youth and Sports project No. LTAIN19087 "Multi-linguality in speech technologies".

3. ENHANCED ASR-TTS (EAT)

The EAT proposed in this work includes two main modifications to our previous work [4], first on SO and other on TO training in the ASR-TTS model.

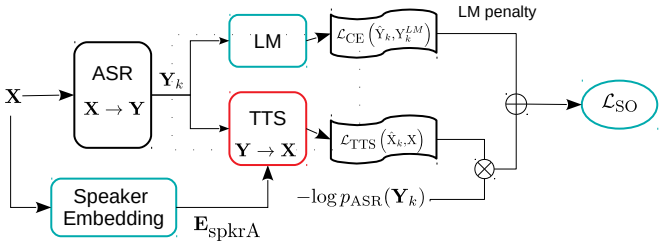

Fig. 1. Speech Only (SO) data training using ASR → TTS pipeline with LM penalty.

3.1. Adding a RNNLM penalty for regularization

The ASR→TTS cycle-consistent SO training objective used in [4]

$$\mathcal{L}'_{\text{SO}} = E_{p_{\text{ASR}}(\mathbf{Y}|\mathbf{X})}\{\mathcal{L}_{\text{TTS}}(\mathbf{X} \mid \mathbf{Y})\}, \quad (1)$$

is the expected TTS negative log-likelihood $\mathcal{L}_{\text{TTS}}(\mathbf{X} \mid \mathbf{Y})$ for the latent ASR hypotheses $\mathbf{Y}$. Note that this likelihood is teacher-forced i.e. the ground truth is used for the auto-regressive component. One limitation of this approach is that cycle-consistency may not be restrictive enough to avoid erroneous hypotheses for $\mathbf{Y}$, making training less robust. To solve this, we incorporate a β-weighted negative log likelihood of a RNN language model to equation (1) as shown in figure 1 and yielding

$$\mathcal{L}_{\text{SO}} = E_{p_{\text{ASR}}(\mathbf{Y}|\mathbf{X})}\{\mathcal{L}_{\text{TTS}}(\mathbf{X} \mid \mathbf{Y}) + \beta\mathcal{L}_{\text{LM}}(\mathbf{Y})\}, \quad (2)$$

which plays a regularization role similar to the Kullback-Leibler term in Variational Auto-encoders (VAEs) [14][1]. The expectation is approximated with REINFORCE [15].

3.2. Making TTS→ASR robust to out-of-domain

The TTS→ASR cycle-consistent TO training objective from [4] exhibits a major weakness when training with out-of-domain data. TTS is less robust to out-of-domain data and generates poor log-Mel filterbank (fbank) frames in this condition. In this pipeline, features $\hat{\mathbf{X}}$ are predicted by TTS as

$$\hat{\mathbf{X}} = \arg\max_{\mathbf{X}}\{p_{\text{TTS}}(\mathbf{X} \mid \mathbf{Y})\} \quad (3)$$

encoded in the ASR encoder as $H = \text{Encoder}(X)$, and sent to the attention component to obtain the attention context vector c_l as

$$c_l = \sum_t a_{lt} h_t \quad (4)$$

[1]Note the similarity with the Evidence Lower Bound (ELBO) $\log p(x) \geq E_{q(y|x)}\{\log p(x \mid y) + \log p(y) - \log q(y \mid x)\}$

where t and l denote the time step and token id respectively. The final loss is then given by

$$\mathcal{L}_{\text{TO}} = -\log p_{\text{ASR}}(\mathbf{Y}^* \mid \hat{\mathbf{X}}) = -\sum_l^L \log \text{Decoder}(c_l, y_{l-1}) \quad (5)$$

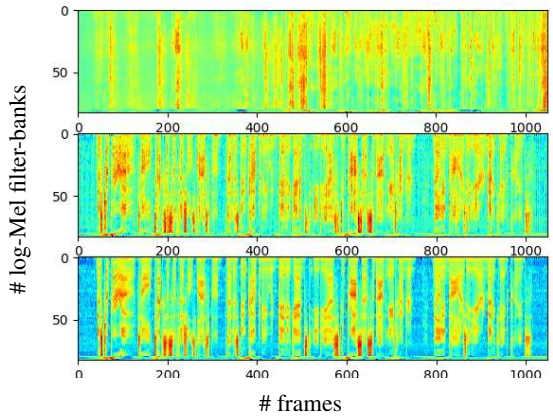

Fig. 2. Plot of log-Mel filter-bank features. The top plot shows the features predicted by TTS during TTS→ASR training. The plot in the middle shows the features predicted by TTS during ASR→TTS training. The bottom figure shows the ground-truth of the log-Mel filterbank (fbank) features.

The top plot in figure 2 shows the reconstructed fbank features of Librispeech using a TTS pre-trained with WSJ data. Comparing the top and middle plots, corresponding TTS→ASR and ASR→TTS training respectively, one can see a clear difference in prediction error.

The primary reason behind this is that the ground truth is available in ASR→TTS to perform teacher-forcing. Whereas in TTS→ASR, this is not available for TTS and thus the reconstructed output deviates from the ground truth. Even the segments of speech and silence are wrongly predicted in the TTS→ASR pipeline.

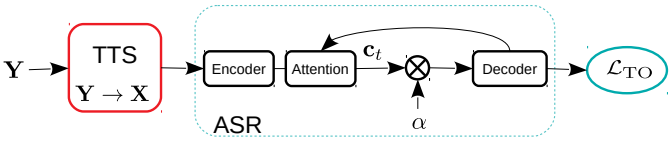

Fig. 3. Text only (TO) data training with inclusion of attention context scaling by α in TTS→ASR pipeline

In EAT, we mitigate this robustness issue by scaling the attention context vector in the ASR module by a hyper-parameter α

$$c_l = \alpha \sum_t a_{lt} h_t \quad (6)$$

also shown in figure 3. If $\alpha = 0$, no encoder features are used and the ASR model just behaves as a language model. This prevents the erroneous TTS generated features to provide a misleading signal, while still allowing to backpropagate into the ASR decoder. The value of α is chosen heuristically based on the difference in domains between data used to train TTS and the TO data. The final loss, used both for speech only and text only data (ST), is given by summing the loss functions $\mathcal{L}_{\text{ST}} = \mathcal{L}_{\text{SO}} + \mathcal{L}_{\text{TO}}$ of the above pipelines.

3.3. Improvements in architecture and training

3.3.1. Model Architecture

The ASR and TTS architecture in the EAT model is meticulously designed as it plays a major role in attaining improved performance. The motivation is to keep the model light-weight and simple to easily fit in GPU memory during training.

ASR: The ASR component in EAT model is equipped with location based multi-head attention component [16]. Instead of RNN layers in encoder, self-attention layers are used to reduce model complexity. The decoder is built with RNN layers as before, since the transformer decoder is harder to implement with our training objective. In our experiments, we use 2 VGG layers followed by 6 self-attention layers each with 800 dimensions. The encoder output is sent to attention component with 10 heads and 512 dimensions. 10 convolution channels with 100 filters are used in this attention to be location specific. Adadelta is used and trained with batch size 20. Our experiments shows that multi-head attention and self-attention layer based encoder provided performance gains.

TTS: Transformer based TTS [17, 18] is used in this work, as we found in our experiments that Transformer consumes less memory and is effective in out-of-domain condition when compared to Tacotron architecture [17]. The TTS is multi-speaker based and handles each speaker input by providing an x-vector [19] as speaker embedding. The Transformer architecture contains 6 encoder and decoder layers each with 1536 units respectively. The attention component contains 4 attention heads, each with 384 attention dimension. 2 pre-net layers with 256 units and 5 post-net layers with 256 channels are used. The output frames reduction factor is set to 1 as all frames are required during self-supervised training. Speaker embeddings are added to the encoder output before sending to the decoder. The pre-training of transformer TTS is done using its standard optimizer with 10000 warmup steps for 100 epochs.

3.3.2. Data Augmentation

In ASR-TTS, simple Gaussian noise is used as augmentation to stabilise the training and it provided minor gains but was inconsistent across datasets. In this work, the EAT model is trained with the specaugment [20] approach. The frequency mask and time mask are applied using a window width of 30 consecutive log-Mel frequency channels and 40 consecutive time steps respectively. The recognition performance of the EAT model with specaugment is shown in table 2. Specaugment approach attains consistent gains using self-supervised training with SO and ST (SO + TO) data. No augmentation is done during training with TO data.

```
in general investors are a conservative lot
these days she says (ground-truth)
in general investors are a conservative lat
these days she says (baseline)
in deneral investors are a conservative lot
these days she say (ASR-TTS)
```

Fig. 4. An example of text sequence predicted by baseline and ASR-TTS compared with the ground-truth

3.3.3. Data Annealing

In ASR-TTS model training, alternating between large amounts of unsupervised data and little supervised data is difficult. The supervised training of certain labels can result in over-fitting, which hinders the effect of unsupervised training [21] as shown in figure 4.

Here, "general" in reference text is correctly predicted in baseline training. During ASR-TTS training, the supervised samples from baseline and unsupervised samples are provided in alternate fashion. Although, in [4] we repeated the supervised data to reduce the underfitting, it still resulted in incorrect predictions such as "deneral" and also increased the training time due to repetition. To mitigate this, the supervised samples are released only when:

$$p_{\text{ASR}}(\hat{y} \mid x) > \gamma_t; \gamma_t = \eta_t \times (1 - \frac{1}{K}) + \frac{1}{K} \times \eta_t \quad (7)$$

$$\eta_t \rightarrow 1 - \exp(\frac{t}{T} * 5), \quad \text{log schedule} \quad (8)$$

$$\eta_t \rightarrow \exp((\frac{t}{T} - 1) \times 5), \quad \text{exp schedule} \quad (9)$$

where T and K are the number of training steps and classes respectively. Table 1 shows that linear and exp schedules are better over log, as release of supervision is initially high and reduces at the end of training. The performance of EAT trained using exp schedule outperforms linear as the supervised data is mostly released at the final stage of training paving smoother way for training with unsupervision.

4. RESULTS AND DISCUSSION

Librispeech [22] and BABEL-Pashto [23] datasets are used in our experiments. The WSJ-si84 is used to pre-train ASR and TTS models. 83 dimensional filterbank features are extracted and used to train our ASR and TTS systems. EAT model training is performed under SO, TO and ST condition by splitting the data into unpaired and paired data. 100 hours of Librispeech is used as paired data and 360 hours of Librispeech as unpaired data. 5 and 10 hours of paired data are obtained from 39.74 hours of BABEL-Pashto data and the rest of the dataset is used as unpaired data as denoted in table 3. Baseline models denoted in table 2 and 3 are built only with paired data. RNNLM for Librispeech is built with 460 hours of clean and 500 hours of other data. RNNLM for Pashto is built with external text containing 83k utterances and 62k vocabulary size. Our experiments are done using ESPnet toolkit and the code will be published on github [2]. All experiments are conducted with RNNLM during testing. Evaluation with Librispeech is done on dev-clean, dev-other, test-clean and test-other as such variability can showcase the effectiveness of EAT.

4.1. Results on Librispeech

EAT is initially tested with different data annealing schedules to chose the best training schedule for the rest of the experiments. 360 hours of both speech only and text only (ST) data is used during EAT training and the results for log, linear and exp schedules are in table 1. The results show that for Librispeech, exp schedule is better and will be used in the rest of our experiments.

Table 1. %WER performance of log, linear and exp based annealing schedules data during self-supervised training using EAT

360-ST	dev-clean	dev-other	test-clean	test-other
log	7.7	23.5	6.9	24.3
linear	7.1	22.7	6.9	23.6
exp	6.9	22.5	6.9	22.1

Table 2 shows the effect of data augmentation by specaugment. SO training with 360 hours of data attains consistent gains on all evaluation sets. 360-ST denotes that the 360 hours of speech only and text

[2] https://github.com/creatorscan/espnet-asrtts

only data are used simultaneously for training EAT. The performance improvements obtained by 360-SO and 360-ST with augmentation shows that the EAT model training is complementary to specaugment approach. Here oracle in the table denotes the performance of ASR model trained with 460 hours of data in Librispeech. The effect of SO and ST data training with RNNLM penalizer (LMP) in EAT is also shown in table 2. The penalty loss aids the model training to reduce incorrect ASR predictions and the results show significant improvement in performance. 360-SO + LMP attains 19.0%WER on harder evaluation set such as test-other when compared to 25.0%WER with specaugment only. The attention context vector in ASR is scaled

Table 2. Recognition performance of EAT model using specaugment approach, RNNLM penalizer (LMP) and attention context scaler (α)

Type	dev-clean	dev-other	test-clean	test-other
Baseline	14.3	36.4	14.4	36.9
360-SO	11.0	32.4	10.6	33.6
+aug	9.1	24.2	8.9	25.0
+LMP	6.0	18.6	5.8	19.0
360-TO	8.9	23.0	8.6	24.1
+α	4.5	15.8	4.7	15.9
360-ST	6.9	22.5	6.9	23.6
+aug	6.0	18.6	5.8	19.0
+α	5.2	19.5	5.3	20.4
+LMP	4.3	14.9	4.3	15.2
oracle	3.7	12.3	3.5	12.6

by $\alpha = 0.7$ for Librispeech and attains 15.9%WER on test-other with 360-TO when compared to 24.1%WER by 360-TO without α scalar. This simple, yet effective method has allowed 360-ST to further improve its performance which shows that LMP and α scaler are complementary. In 360-ST, inclusion of α and LMP results in 4.3% WER on test-clean and 14.9% WER on test-other. The oracle experiment is done by training an ASR with 460 hours of supervised data and it attains 3.5%WER on test-clean and 12.6%WER on test-other.

Table 3. %WER of EAT on BABEL-Pashto using aug, α for TO and LMP for SO with 5 and 10 hours of paired data

Supervision info	Baseline	SO	TO	ST
5 hours	71.4	63.9	62.2	63.1
+ aug	63.1	61.4	61.7	60.1
+ α	-	-	58.8	58.5
+ LMP	-	60.1	-	58.4
10 hours	64.7	61.1	60.9	60.4
+ aug	55.5	55.0	54.0	53.6
+ α	-	-	53.7	52.5
+ LMP	-	54.8	-	51.6
Oracle	56.0	-	-	-
+ aug	48.9	-	-	-

4.2. Results on BABEL-Pashto

The key results of the EAT are shown on BABEL-Pashto as it helps to show the impact of α and LMP when compared to our previous ASR-TTS model. Experiments on Pashto using our previous model did not lead to gains and hence are not included in this paper. The reason behind the difficulty is that building a multi-speaker TTS model for Pashto is harder and hence our previous work failed to provide reasonable TTS scores. EAT model mitigates this problem by modifying its TTS architecture and reducing the importance of synthesized speech from TTS by α. Here, the pre-trained TTS is retrained during SO with RNNLM penalizer and later used for TO training with $\alpha = 0.3$. Table 3 shows that with 5 hours of paired data, the effect of TO is higher compared to SO, but with 10 hours the TO obtains comparable gains as SO. With α in TO the model obtains 58.8 %WER and further reduced to 58.5 %WER with ST training. Here oracle in table denotes the performance of ASR trained with 39.75 hours of data in Pashto. With 10 hours of paired data and ST training with both α and LMP, the EAT model attains 51.6 %WER which is only absolute 2.7% less compared to oracle 48.9 %WER.

4.3. Comparison with related work

Some of the recent works using ASR and TTS to handle unpaired speech and text data have raised the performance bar on Librispeech. The self-training approach such as pseudo-label training [24] feeds the SO data to a pre-trained ASR and uses the predicted hypotheses as pseudo-labels to attain better performance on all dev and test sets. The errors in pseudo-labels were further corrected with a language model using local prior matching (LPM) objective [25] and led to improvements in dev-other and test-other while the performance slightly degrades in dev-clean and test-clean. Our EAT model outperforms the self-training model on all evaluation sets as noted in table 4.

Table 4. Comparison of SotA results in literature with EAT model

Method	Type	dev		test	
		clean	other	clean	other
Self-training	Pseudo [10]	5.41	20.31	5.79	21.63
	LPM [25]	5.69	20.22	5.99	20.93
Synthesis	GST [7]	7.4	25.7	7.9	26.7
	GCP [26]	4.1	-	4.1	-
Cycle	ASR-TTS	11.0	32.4	10.6	33.6
	EAT	4.3	14.9	4.3	15.2

GST [7] method focuses on attaining better synthesis quality by using GST speaker embeddings by training with TO data which is further used to train an ASR. This model attains 7.4%WER and 7.9%WER on dev-clean and test-clean which is relatively less due to the small language model used. In case of GCP, the authors synthesize speech and use consistency loss together to attain 4.1%WER on dev-clean and 4.1%WER on test-clean. This is understandably better than our EAT model since the GCP uses 460 hours of paired data while we use only 100 hours of paired data. The effect of penalizer, attention context scalar and other training strategies makes our EAT model attain 4.3%WER on dev-clean and 4.3% on test-clean. The model also attains the best performance on harder conditions such as dev-other and test-other.

5. CONCLUSION

In this work, we address the shortcomings of our previous ASR-TTS model and propose an enhanced ASR-TTS model. The proposed EAT model performs well on commonly used Librispeech task and shows its robustness to domain changes on BABEL-Pashto. The modification of SO with penalizer helped the model to improve on language related errors without hurting the acoustic information captured. The TO training by scaling attention context helped to improve on out-of-domain conditions such as Pashto and also brought gains in Librispeech. Training speech and text only (ST) together proved to be complementary and resulted in further performance improvement. The performance of EAT on Pashto is 51.6%WER which is 2.7% absolute less than with oracle 48.9%WER. The model also attains 15.2%WER on test-other which is 2.6% absolute less than oracle performance. The model can be further enhanced on Librispeech by using 960 hours of unpaired speech and text data.

6. REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

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

[3] 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.

[4] M. K. Baskar, S. Watanabe, R. Astudillo, T. Hori, L. Burget, and J. Černocký, "Semi-supervised sequence-to-sequence ASR using unpaired speech and text," *arXiv preprint arXiv:1905.01152*, 2019.

[5] 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*, pp. 426–433, 2018.

[6] A. Tjandra, S. Sakti, and S. Nakamura, "End-to-end feedback loss in speech chain framework via straight-through estimator," *arXiv preprint arXiv:1810.13107*, 2018.

[7] N. Rossenbach, A. Zeyer, R. Schlüter, and H. Ney, "Generating synthetic audio data for attention-based speech recognition systems," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7069–7073, IEEE, 2020.

[8] G. Sun, Y. Zhang, R. J. Weiss, Y. Cao, H. Zen, A. Rosenberg, B. Ramabhadran, and Y. Wu, "Generating diverse and natural text-to-speech samples using a quantized fine-grained vae and autoregressive prosody prior," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6699–6703, IEEE, 2020.

[9] C. Du and K. Yu, "Speaker augmentation for low resource speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7719–7723, IEEE, 2020.

[10] J. Kahn, A. Lee, and A. Hannun, "Self-training for end-to-end speech recognition," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020.

[11] A. Baevski, M. Auli, and A. Mohamed, "Effectiveness of self-supervised pre-training for speech recognition," *arXiv preprint arXiv:1911.03912*, 2019.

[12] 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.

[13] R. Masumura, M. Ihori, A. Takashima, T. Moriya, A. Ando, and Y. Shinohara, "Sequence-level consistency training for semi-supervised end-to-end automatic speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7054–7058, IEEE, 2020.

[14] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[15] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[16] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, *et al.*, "A comparative study on transformer vs RNN in speech applications," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 449–456, IEEE, 2019.

[17] T. Hayashi, R. Yamamoto, K. Inoue, T. Yoshimura, S. Watanabe, T. Toda, K. Takeda, Y. Zhang, and X. Tan, "Espnet-tts: Unified, reproducible, and integratable open source end-to-end text-to-speech toolkit," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7654–7658, IEEE, 2020.

[18] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, "Neural speech synthesis with transformer network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 6706–6713, 2019.

[19] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *ICASSP*, pp. 5329–5333, 2018.

[20] 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," *arXiv preprint arXiv:1904.08779*, 2019.

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

[22] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *ICASSP*, pp. 5206–5210, 2015.

[23] M. Karafiát, M. K. Baskar, P. Matějka, K. Veselý, F. Grézl, and J. Černocký, "Multilingual BLSTM and speaker-specific vector adaptation in 2016 BUT Babel system," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*, pp. 637–643, IEEE, 2016.

[24] G. Synnaeve, Q. Xu, J. Kahn, E. Grave, T. Likhomanenko, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, "End-to-end ASR: from supervised to semi-supervised learning with modern architectures," *arXiv preprint arXiv:1911.08460*, 2019.

[25] W.-N. Hsu, A. Lee, G. Synnaeve, and A. Hannun, "Semi-supervised speech recognition via local prior matching," *arXiv preprint arXiv:2002.10336*, 2020.

[26] 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 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7029–7033, IEEE, 2020.