Interspeech 2025
17-21 August 2025, Rotterdam, The Netherlands

EmbedAug: An Augmentation Scheme for End-to-End Automatic Speech Recognition

Ashish Panda[1], *Sunil Kumar Kopparapu*[1]

[1]TCS Research, Tata Consultancy Services, Mumbai, India
{ashish.panda, sunilkumar.kopparapu}@tcs.com

Abstract

Data augmentation plays a significant role in making automatic speech recognition (ASR) systems robust against unseen test data. Most of the existing data augmentation techniques are designed to work on the speech features. Augmentation of speech embeddings, within the neural network, e.g., the inputs to encoders, are relatively unexplored. We present a simple yet effective augmentation scheme, EmbedAug, which works by replacing a set of randomly selected speech embeddings by either zeros or Gaussian noise during training. EmbedAug does not require additional data, works online during training and adds very little to the overall computational cost. Using Librispeech 100h, Librispeech 960h and MUCS21 multilingual dataset, we show that the proposed EmbedAug is very effective in improving the robustness of ASR systems. Moreover, EmbedAug can be fine tuned on the development set with the help of just one hyperparameter.
Index Terms: speech recognition, augmentation, robustness

1. Introduction

End-to-end automatic speech recognition (ASR) systems, such as Conformer [1] based ASR systems, have set new benchmarks for performance in the past few years. They perform very well when there is abundance of training data. However, they do not perform as well in low resource conditions [2]. Data augmentation is a common strategy to increase the amount of training data [3]. There have been numerous studies addressing the data augmentation policies for ASR systems [4], most of them working either on raw audio data or on the feature vectors extracted from the raw audio. Literature relating to augmentation in ASR is vast and we review a part of it below.

Speed Perturbation proposed in [5], creates multiple instances of the speech signal which differ in the speed. Speed perturbation accounts for the variability in speaking rate, which can occur in natural speech. Vocal Tract Length Perturbation (VTLP) [3] transforms speech data by linearly warping the spectrograms of input audio utterance by a random warp factor. VTLP accounts for the speaker variability in speech. In [6], different versions of relatively clean data are created for different room dimensions and different reverberation time. This way, it accounts for speech spoken in different reverberant conditions. For noise robust ASR, training speech signals are corrupted by artificial noise in [7]. Recently, SpecAugment [8] has received a lot of attention. It transforms the mel-spectrogram features of a speech signal and has three components: time warp, time mask and frequency mask and provides remarkable improvement in ASR performance. A variation of it, SpecSwap, was proposed in [9], but it slightly underperformed SpecAugment with time-warp component deleted.

While data augmentation at feature level has become standard in many ASR systems, embedding augmentation has not received much attention. Another short-coming of SpecAugment is that it has 7 hyperparameters for fine tuning, which require many experiments to fix the optimal values. We try to address these two issues in this paper. After summarizing the related work in Section 2, the remainder of the paper is organized as follows. Section 3 introduces the proposed augmentation scheme EmbedAug. Section 4 presents the experimental results and analysis, which show the efficacy of EmbedAug. Section 5 concludes the paper and outlines some of the future work.

2. Related Work

2.1. Dropout related works

Dropout [10] is a regularization scheme which works by randomly dropping nodes/units along with their connections during training. During inference, the weights are scaled to account for the dropped units. A variation of this is DropConnect [11], which sets a randomly selected subset of weights within the network to zeros. While dropout and DropConnect drop or mask a randomly selected subset of weights in the network, LayerDrop [12] and stochastic depth [13] extend this to entire layers. The idea behind these dropout variations is that the trained network is rendered robust if a subset of the network weights/units are ignored during training. InterAug [14] (not to be confused with its homonym InterAug for image processing [15], which uses the bounding box annotation effectively) implements SpecAugment like time masking and frequency masking in the hidden layer outputs. Therefore, it can be thought of as incorporating some aspects of LayerDrop and DropConnect to improve low resource ASR performance. However, InterAug does not augment the speech embeddings (inputs to encoders).

2.2. Embedding masking related works

Self-supervised and semi-supervised approaches for ASR such as HuBERT [16] and Wav2Vec 2.0 [17] use a scheme for masking the output of the convolution layer during pre-training. This forms the input to the transformer encoder, which learns the speech representations. Both HuBERT and Wav2Vec 2.0 scheme randomly sample $p\%$ indices, without repetition, from the set of embedding indices for each utterance. The sampled indices form the start indices of masked blocks, where each block consists of M indices. The loss function during pre-training is computed as the weighted sum of loss computed over the masked timesteps and the loss computed over unmasked timesteps. Temporal Dropout scheme [18] uses a similar strategy to drop a contiguous set of video frames, but does not ac-

10.21437/Interspeech.2025-1360

count for dropped frames in the loss function. The masking scheme in BERT pretraining [19] advocates masking 15% of the tokens at random. It should be noted that the masking scheme proposed for BERT, HuBERT and Wav2Vec 2.0 are used during pre-training and not in an end-to-end set-up.

2.3. Embedding Masking in End-to-end ASRs

To the best of our knowledge, augmentation of speech embeddings that form input to the encoder module has not been explored for end-to-end ASR training. Although HuBERT and Wav2Vec 2.0 have implemented contiguous speech embedding augmentation in pre-training, it does not follow that the same scheme will work for end-to-end ASR training. In fact, we show that the masking scheme used in HuBERT and Wav2Vec 2.0 does not work as well for end-to-end ASRs. Also, almost all masking schemes in literature propose masking of consecutive blocks of time-steps. However, no reason has been provided as to why masking of a block of time-steps might be beneficial as compared to masking random embeddings. We compare the two approaches of masking in this paper.

We propose EmbedAug, in the next section, which differs in two important ways. First, EmbedAug masks a certain percentage of randomly selected speech embeddings. Second, it uses two different masking approaches, i.e., masking by zeros and masking by Gaussian noise, $\mathcal{G}(\mu = 0, \sigma^2 = 1)$ with mean zero and variance of one.

3. EmbedAug

The motivation for this scheme stems from the fact that speech signal is hardly ever the same, even when the same speaker speaks the same text on two different occasions within a short span of time. This would imply that the exact sequence of embeddings may not be very important. Therefore, the goal of EmbedAug is to force the network to learn the prediction without seeing the exact sequence of the embeddings by masking a randomly selected set of speech embeddings. However, the key questions are whether masking of *contiguous bands* of embeddings works for an end-to-end ASR, what *percentage* of the speech embeddings should be masked, and what should the *mask value* be? We experiment with the following variations to answer these questions.

We randomly select $p\%$ of the embeddings, without repetition, and mask them. This scheme simulates smaller chunks of missing data at the embedding level. Contiguous bands can still occur with this scheme when the selected embeddings are adjacent to each other. Note that while the minimum band size is M with masking of contiguous bands (e.g. Wav2Vec2.0 style masking), the minimum band size with our scheme is 1. In other words, masking of contiguous bands precludes masking of single embeddings while this scheme does not preclude masking of contiguous bands. Optimal performance can be arrived at by varying just one parameter p.

Figure 1 illustrates our approach of speech embedding augmentation. Now, we come to masking value. We experiment with three variations. In one variation, masking value can be zeros, i.e., the masked embeddings are set to a very small number (e.g. 10^{-6}). Zeros simulate missing features and therefore they can simulate deletion errors of ASR systems. In another variation, the masked embeddings are replaced with Gaussian noise with zero mean and unit variance. Gaussian noise can simulate noisy embeddings, which can make the network robust. In yet another variation, each utterance has a 50% chance

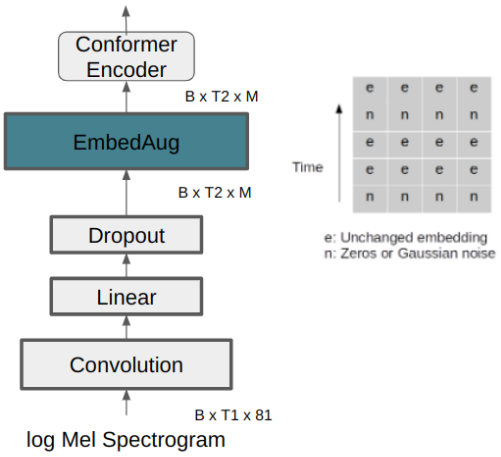


Figure 1: *Proposed Augmentation Policy for Speech Embeddings. B is the mini batch size, T1 is the number of frames, T2 is the number of the embeddings in an utterance and M is the model dimension.*

of being masked by zeros and 50% chance of being masked by Gaussian noise. This way, the network sees both variations of the embeddings. Seeing both variations can potentially make the network robust against both missing and noisy features.

Suppose T_2 is the number of embeddings for an utterance. Let $I = \{t_1, t_2, ..., t_k\}$ be the indices of randomly selected embeddings to be masked, where $k = \text{floor}(p/100) * T_2$ and r is randomly chosen to be 0 or 1 (r remains unchanged for an utterance), then mathematically, EmbedAug can be described as:

$$e(t,m) = \begin{cases} 0, & r = 1 \\ n, & r = 0 \end{cases} \quad t \in I, \forall m \tag{1}$$

In the above equation, $e(t, m)$ stands for a speech embedding, t stands for the temporal index and m stands for model dimension index and $n \sim \mathcal{G}(\mu = 0, \sigma^2 = 1)$.

It can be seen that the proposed policy for EmbedAug can be done online during training with negligible computational overhead and it does not create any additional data. Therefore, it shares all the computational advantages of SpecAugment. It can also be seen how the proposed scheme is different from LayerDrop and stochastic depth methods. While EmbedAug proposes to mask $p\%$ of embeddings from an utterance, LayerDrop and stochastic depth drop an entire layer for the whole of a mini batch. EmbedAug is performed only during the training phase and for the encoder only. The sub-word embeddings for the decoder are not augmented. Another thing to note is that EmbedAug is used along with dropout.

4. Experiments

We have conducted experiments on the Librispeech database [20] and MUCS-21 database [21]. We have implemented conformer encoder and transformer decoder architecture for the ASR tasks, using ESPNet toolkit [22] with ESPNet-1 recipes. For Librispeech 100h and MUCS-21 subtask-1 training, the model dimension was 256 with 4 attention heads. For Librispeech 960h training, the model dimension was 512 with 8

attention heads. A single Nvidia RTX 3090 GPU was used. All experiments for each dataset were conducted on the same machine so that the performance comparisons are fair. A set of 300 sub-word tokens extracted from the training text served as output units for Librispeech 100h and MUCS21 training, while 5000 subword tokens were used for Librispeech 960h training. Features used for the model are log mel spectrograms with 80 dimensions along with the pitch (total 81, see Figure 1). Speed Perturbation was not used for Librispeech 960h and MUCS-21 tasks in order to reduce training time. Batch size of 36 was used for Lirbispeech 960h while 64 was used for Librispeech 100h and MUCS-21 dataset so that GPU memory is optimally used.

4.1. Librispeech 100h

First set of experiments were conducted on the low-resource Librispeech 100h training set. Table 1 summarizes the word error rates (WERs) from different experiments. The baseline system with SpecAugment performs well with WERs for test-clean and test-other at 8.1% and 20.0% respectively, resulting in an average WER of 14.0 % on the test sets (namely test-avg).

Next, we experiment with the contiguous embedding masking scheme described in Section 2.2. The recommended values of p and M are 6.5 and 10 respectively according to [17]. When we use these values, the system becomes erratic resulting in test-avg WER of 168.5%. Although this masking scheme has been shown to work for Wav2Vec 2.0 pre-training, it does not seem to be working for end-to-end ASR. Only when we reduce M to 2, do we see better performance with the best test-avg WER obtained for $p = 15$ and $M = 2$ (14.9%), which is still not close to SpecAugment.

Next we look at the EmbedAug scheme of masking randomly selected embeddings, where p% of the speech embeddings were replaced with zeros. Mixed masking scheme, with randomly selected 60% ($p = 60$) of embeddings masked with either zeros or $\mathcal{G}(0, 1)$ noise with equal probability, performs the best overall in dev sets as well as on the test sets. Using the same masking scheme to mask log-mel-spectrum of the speech does not work as well as using it on the speech embeddings (Results are listed in Table 1 with the heading of SpecMask). Using joint CTC-Attention multi-task learning with CTC loss weight 0.3 [23], EmbedAug(Mix, $p = 60$) significantly outperforms SpecAugment reducing the test-avg WER from 13.7% to 12.8%. In this set of experiments, no shallow fusion with a Language Model (LM) was done since ESPNet-1 recipe does not call for it.

4.2. Librispeech 960h

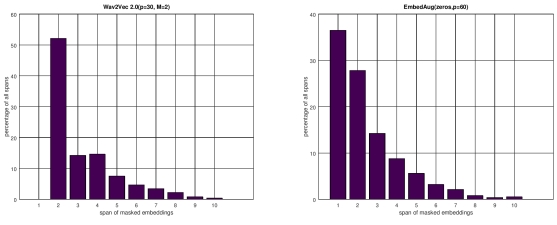
The above set of experiments show that EmbedAug(Mix, $p = 60$) results in best performance. The next set of experiments were done on Librispeech 960h training set. Here joint CTC-attention multi-task learning is used (with CTC loss weight = 0.3), since the above experiments show its advantage. We can see from Table 2 that EmbedAug reduces the WER on the test-avg by an absolute 0.5% (relative 9.2%) as compared to SpecAugment. When a transformer based language model (downloaded from from [24]) was used for shallow fusion, EmbedAug reduced the WER on test-avg by absolute 0.3% (relative 7.6%) compared to SpecAugment. It should be noted that the results for Librispeech 960h reported here are better than the ESPNet reported results on Librispeech for ESPNet-1 recipes without Speed Perturbation[1].

4.3. MUCS-21

The next set of experiments were conducted on the MUCS-21 [21] multilingual dataset. A single model was trained using training data from all languages. For this dataset, we investigated various values of p on the test set and the best performing p was used on the blind set. The results of the experiments are summarized in Table 3. From the results, it is clear that EmbedAug(Mix, $p = 20$) performs the best in both test and blind set. In blind set results, we have not reported the results for Marathi language, since the difference in channel compression for Marathi data between train and blind set leads to spurious results with multilingual end-to-end ASRs [21]. Shallow fusion with a transformer LM, trained on multilingual text, resulted in an EmbedAug WER of 27.7% as compared to SpecAugment WER of 28.0%. (WER for blind set Marathi data with SpecAugment was 100.8%, while the same for EmbedAug(Mix, $p = 20$) was 75.4%.)

4.4. Discussion

We posited that masking scheme and the mask value are key to obtaining good performance. The experimental results support this too. Figure 2 plots the histograms for masked span of embeddings for Wav2Vec(p = 30, M = 2) and EmbedAug(zeros, $p = 60$) for 10 utterances. Both schemes mask same proportion of embeddings, i.e., 60%. The difference lies in number of contiguous embeddings being masked. As we have stated in Section 3, the contiguous scheme of masking precludes individual embeddings being masked. Number of consecutive embeddings getting masked is also smaller for EmbedAug. That seems to bring about the vast difference in the performance of the ASR systems.



(a) (b)

Figure 2: *Mask span distribution for 10 utterances. (a) Contiguous(p=30,M=2), (b) EmbedAug(zeros,p=60).*

5. Conclusion

In this paper, we presented a simple but effective technique for speech embedding augmentation which improves performance of end-to-end ASR systems. The proposed augmentation technique outperforms SpecAugment on Librispeech 100h, Librispeech 960h, and MUCS-21 datasets. The gain in performance was achieved without much computational cost. There is only one parameter in the EmbedAug scheme, p, which can be varied to obtain the best performance. The optimal value of this parameter can be arrived at with the help of development data. This work shows that embedding masking is a worthwhile topic for exploration and that the scheme of masking and the choice of mask values are of critical importance. In future, We propose to experiment with other probabilities for zeros and Gaussian noise.

[1] https://github.com/espnet/espnet/blob/master/egs/librispeech/asr1/RESULTS.md

Table 1: *WERs (in %) for various schemes on Librispeech 100h. "dev-avg": average over dev-clean and dev-other. "test-avg": average over test-clean and test-other*

Masking Scheme	dev-clean	dev-other	dev-avg	test-clean	test-other	test-avg
SpecAugment	7.5	19.5	13.5	8.1	20.0	14.0
Wav2Vec 2.0 ($p = 6.5$, $M = 10$)	160.4	174.5	167.4	161.2	175.9	168.5
Wav2Vec 2.0 ($p = 6.5$, $M = 5$)	183.7	200.4	192.0	183.8	201.8	192.8
Wav2Vec 2.0($p = 6.5$, $M = 2$)	8.0	23.1	15.5	8.7	24.0	16.3
Wav2Vec 2.0 ($p = 10$, $M = 2$)	8.9	23.9	16.4	9.2	24.5	16.8
Wav2Vec 2.0 ($p = 15$, $M = 2$)	7.8	21.7	14.7	8.0	21.9	14.9
Wav2Vec 2.0 ($p = 20$, $M = 2$)	9.4	24.2	16.8	10.0	24.8	17.4
Wav2Vec 2.0 ($p = 30$, $M = 2$)	160.9	174.6	167.7	161.1	176.4	168.7
EmbedAug (zeros, $p = 50$)	8.7	23.1	15.9	9.4	23.8	16.6
EmbedAug (zeros, $p = 60$)	7.1	20.2	13.6	7.3	20.1	13.7
EmbedAug (zeros, $p = 65$)	7.5	20.0	13.7	7.9	20.4	14.1
EmbedAug ($\mathcal{G}(0, 1)$, $p = 50$)	7.6	20.5	14.0	7.7	20.9	14.3
EmbedAug ($\mathcal{G}(0, 1)$, $p = 60$)	7.7	20.1	13.9	8.0	20.2	14.1
EmbedAug ($\mathcal{G}(0, 1)$, $p = 65$)	7.4	20.7	14.0	7.8	21.1	14.4
EmbedAug (Mix, $p = 60$)	7.3	19.4	13.3	7.7	19.8	13.7
SpecMask (Mix, $p = 40$)	8.7	21.5	15.1	9.2	21.6	15.4
SpecMask (Mix, $p = 60$)	8.6	21.0	14.8	8.8	20.9	14.8
SpecMask (Mix, $p = 80$)	9.5	21.4	15.4	9.7	22.0	15.8
SpecAugment+CTC-Att	7.4	19.2	13.3	7.7	19.7	13.7
EmbedAug (Mix, $p = 60$) + CTC-Att	6.8	19.3	**13.0**	6.9	18.8	**12.8**

Table 2: *WERs (in %) for various schemes on Librispeech 960h. "dev-avg": average over dev-clean and dev-other. "test-avg": average over test-clean and test-other. "LM": shallow fusion with language model*

Masking Scheme	dev-clean	dev-other	dev-avg	test-clean	test-other	test-avg
SpecAugment	2.9	7.6	5.2	3.2	7.6	5.4
EmbedAug (Mix, $p = 60$)	2.7	6.8	**4.7**	3.0	6.8	**4.9**
SpecAugment+LM	2.2	5.3	3.7	2.4	5.5	3.9
EmbedAug (Mix, $p = 60$)+LM	2.1	4.9	**3.5**	2.2	5.0	**3.6**

Table 3: *WERs (in %) for various schemes on MUCS-21 dataset. WERs reported on test and blind sets. "LM": shallow fusion with language model, "Avg" : average over all languages*

Masking Scheme [decode set]	Gujarati	Hindi	Marathi	Odia	Tamil	Telugu	Avg
SpecAugment [test]	23.8	25.9	19.3	39.4	28.5	30.4	27.9
EmbedAug (Mix, $p = 10$) [test]	24.8	26.4	17.6	36.8	30.5	31.0	27.8
EmbedAug (Mix, $p = 20$) [test]	24.3	25.2	17.8	36.2	29.3	30.0	**27.1**
EmbedAug (Mix, $p = 30$) [test]	24.1	26.1	18.8	37.7	29.0	29.9	27.6
SpecAugment [blind]	35.2	23.1	–	41.2	32.7	36.4	33.7
EmbedAug (Mix, $p = 20$) [blind]	37.4	21.5	–	36.3	33.3	36.2	**32.9**
SpecAugment+LM [blind]	28.6	19.7	–	38.3	26.2	27.4	28.0
EmbedAug (Mix, $p = 20$)+LM [blind]	30.7	18.6	–	33.5	27.5	28.0	**27.7**

6. References

[1] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," *ArXiv*, vol. abs/2005.08100, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:218674528

[2] R. Prabhavalkar, T. Hori, T. N. Sainath, R. Schlüter, and S. Watanabe, "End-to-end speech recognition: A survey," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 32, pp. 325–351, 2024.

[3] N. Jaitly and E. Hinton, "Vocal tract length perturbation (VTLP) improves speech recognition," in *Proc. International Conference on Machine Learning (ICML)*, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:14140670

[4] B. Thai, R. Jimerson, D. Arcoraci, E. Prud'hommeaux, and R. Ptucha, "Synthetic data augmentation for improving low-resource asr," in *2019 IEEE Western New York Image and Signal Processing Workshop (WNYISPW)*, 2019, pp. 1–9.

[5] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. Interspeech 2015*, 2015, pp. 3586–3589.

[6] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. Sainath, and M. Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in google home," in *Proc. Interspeech 2017*, 2017, pp. 379–383.

[7] M. Soni, S. Joshi, and A. Panda, "Generative Noise Modeling and Channel Simulation for Robust Speech Recognition in Unseen Conditions," in *Proc. Interspeech 2019*, 2019, pp. 441–445.

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

[9] X. Song, Z. Wu, Y. Huang, D. Su, and H. Meng, "Specswap: A simple data augmentation method for end-to-end speech recognition," in *Interspeech 2020*, 2020, pp. 581–585.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, p. 1929–1958, jan 2014.

[11] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, "Regularization of neural networks using dropconnect," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1058–1066. [Online]. Available: https://proceedings.mlr.press/v28/wan13.html

[12] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," *ArXiv*, vol. abs/1909.11556, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:202750230

[13] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 646–661.

[14] Y. Nakagome, T. Komatsu, Y. Fujita, S. Ichimura, and Y. Kida, "InterAug: Augmenting Noisy Intermediate Predictions for CTC-based ASR," in *Proc. Interspeech 2022*, 2022, pp. 5140–5144.

[15] D. S, K. Thopalli, and J. J. Thiagarajan, "InterAug: A Tuning-Free Augmentation Policy for Data-Efficient and Robust Object Detection," in *4th Visual Inductive Priors for Data-Efficient Deep Learning Workshop*, 2023. [Online]. Available: https://openreview.net/forum?id=Ole2LywcNw

[16] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 29, p. 3451–3460, oct 2021. [Online]. Available: https://doi.org/10.1109/TASLP.2021.3122291

[17] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *CoRR*, vol. abs/2006.11477, 2020. [Online]. Available: https://arxiv.org/abs/2006.11477

[18] D. Zhang, C. Li, F. Lin, D. Zeng, and S. Ge, "Detecting deepfake videos with temporal dropout 3dcnn," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Z.-H. Zhou, Ed. International Joint Conferences on Artificial Intelligence Organization, 8 2021, pp. 1288–1294, main Track. [Online]. Available: https://doi.org/10.24963/ijcai.2021/178

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[20] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[21] A. Diwan, R. Vaideeswaran, S. Shah, A. Singh, S. Raghavan, S. Khare, V. Unni, S. Vyas, A. Rajpuria, C. Yarra, A. Mittal, P. K. Ghosh, P. Jyothi, K. Bali, V. Seshadri, S. Sitaram, S. Bharadwaj, J. Nanavati, R. Nanavati, and K. Sankaranarayanan, "Mucs 2021: Multilingual and code-switching asr challenges for low resource indian languages," in *Interspeech 2021*, 2021, pp. 2446–2450.

[22] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-End Speech Processing Toolkit," in *Proc. Interspeech 2018*, 2018, pp. 2207–2211.

[23] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," 2017. [Online]. Available: https://arxiv.org/abs/1609.06773

[24] "Pre-trained model," https://drive.google.com/drive/folders/1GFaqEKsI6M_dBbxHD7s84J3yi5WGeNNC, accessed: 2024-07-02.