



Beyond Attacks: Advancing Fake Speech Detection with Attack-Agnostic Methods

Shilpa Chandra¹, Akansha Tyagi¹, Shiven Patel¹, Padmanabhan Rajan¹

¹School of Computing and Electrical Engineering, Indian Institute of Technology, Mandi, India

{s22004,d19030,b22176}@students.iitmandi.ac.in, padman@iitmandi.ac.in

Abstract

Detectors of fake speech are prone to poor performance when presented with data unseen during training. Unseen data can include not only new methods for generating fake speech and various transformations applied to fake speech, but also fake speech in new languages. This study investigates two methods for improving generalization capability of detectors that use a self supervised model as a front-end. The first method uses an encoder-decoder architecture to learn representations robust to different types of fake speech. The second method uses a subspace-based decomposition and learns common information present in various types of fake speech. The proposed methods are evaluated on the standard ASVspoof 2021 dataset, as well as on a multilingual speech dataset of Indic languages. Experiments reveal that state-of-the-art detectors struggle with speech in unseen languages, and the proposed generalization methods help in reducing the performance gap¹.

Index Terms: Fake speech detection, speech spoofing, ASVspoof, domain generalization

1. Introduction

Recent advancements in generating synthetic speech have resulted in the easy creation of speech deepfakes. Research into the detection of speech deepfakes has evolved from the use of hand-engineered features, to the use of representation learning-based techniques. Of late, many researchers apply the use of self-supervised learning (SSL) to provide effective representations for various downstream tasks. SSL models employ large-scale datasets to learn meaningful representations without any manual annotations. The representations thus learned are typically used as front-ends (also called feature extractors) and are used with supervised learning models. In speech, models such as wav2vec 2.0 and its variants, and Hidden-unit BERT (HuBERT) etc. are commonly used as SSL models [1, 2].

As demonstrated by various studies [3, 4, 5], the use of SSL models to detect speech deepfakes has shown promise. Being trained on genuine speech, the SSL model provides a representation which sufficiently discriminates between genuine and fake speech. In addition, the use of powerful classifiers, specifically designed to discriminate genuine and fake speech (such as AASIST [6]) help in providing commendable performance on standard datasets such as ASVspoof 2019 and 2021 [7, 8].

A recurring challenge in the detection of fake speech is the ability to generalize well to unseen types of fake speech (commonly termed as unseen *attacks* in speech anti-spoofing literature.) Studies such as [9] have demonstrated that many detectors

perform poorly in uncontrolled data scenarios. Data augmentation is a commonly used technique to overcome this limitation. Some studies have proposed detectors that use large-scale neural vocoding for data augmentation. Such techniques, though effective, require large amounts of computational power to create or fine-tune the detector. One aspect of fake speech that has been less studied is the ability of classifiers such as AASIST to work on multiple languages. Various research efforts such as Bhashini have resulted in text-to-speech (TTS) systems for closely related language families such as the Indic languages [10]. Thus, one of the objectives of this paper is to study the performance of fake speech detectors on speech generated in Indic languages.

We also explore two methods to improve the generalization ability of fake speech detectors. The motivation is to use relevant information present in the training data and ignoring what is irrelevant. The first technique uses an encoder-decoder framework which learns attack-invariant representations. The second technique uses a subspace-based method to learn a classifier which retains attack-invariant information, and ignores attack-specific information. Hence, we term them *attack-agnostic*. Our belief is that the resulting representations are helpful to ignore aspects such as language, speaker and speech transformations, but retain aspects such as artifacts which result in the production of fake speech.

2. Related Work

Initially, detectors for fake speech relied on using hand-crafted features such as Linear Frequency Cepstral Coefficients, Mel-spectrograms, and similar features, which were then fed into neural networks. Subsequent works introduced more complex architectures, including conformers and graph neural networks [11, 12, 13, 6, 3]. A commonly used graph-based method is AASIST which focuses on identifying spoofing artifacts in the temporal and spectral domains [6].

As models increasingly depended on large datasets, self-supervised learning (SSL) has become a key approach for various speech classification tasks. Recently, wav2vec 2.0 has been widely adopted for detecting fake speech [3]. Guo et al. utilized the more recent WavLM model, which reportedly is better suited for noisy environments [14].

Some approaches also focus on generating additional fake speech to enhance the detection of fake speech. Several approaches have proposed using additional vocoded data to enhance the performance of fake speech detection tasks [15, 16, 17]. These methods suggest that training on more diverse fake speech can improve detection performance.

A few recent papers touch upon multilingual fake speech detection. In [18], the authors propose a dataset of synthetic speech in six languages. In [19], the authors study various

¹Code is available at <https://github.com/shilpac131/AttackAgnostic>

fake speech detectors across multiple languages and conclude that generalization abilities across languages is limited. In [20], the authors create a dataset with synthetic speech from 38 languages and demonstrate that when used for training, the dataset helps to generalize cross-dataset evaluations.

Few works using SSL models rely on using information from all transformer layers [14, 21]. However, recent studies have shown that different layers of SSL models capture distinct properties of phonetic information. Pasad et al. [22] conducted a detailed analysis of which layers in wav2vec 2.0 learn specific acoustic properties. One study provided an in-depth analysis of how various layers perform in detecting fake speech, revealing that the fifth layer of the wav2vec 2.0 large model is effective for this task [4].

In our work, we focus on utilizing the fifth layer of XLS-R which is a variant of the wav2vec 2.0 large model [23], as it captures the most relevant features for our task while minimizing unnecessary complexity. This model is henceforth referred to as w2v2.

3. Attack-Agnostic Methods

In uncontrolled scenarios, the detection of fake speech is challenging, mostly due to (unknown) factors accounting for variability. For example, in synthesized speech, the use of various neural vocoders can provide signal-level deviations. Further, speech may be subject to various transformations, including compression and encoding. An additional unknown factor is the language of the speech. An attack-agnostic detector must be able to capture the characteristics of genuine speech and distinguish them from those of artifacts.

3.1. Attack Invariant Encoder-Decoder (AIED)

Denosing autoencoders [24] have been used successfully in speech enhancement. These networks are trained with noisy-clean pairs with an appropriate loss function. Once trained, the network learns to infer the corresponding clean speech when presented with noisy speech. The bottleneck layer of such a network learns a representation that ignores the noisy component of speech.

Similar to a denosing autoencoder, an attack-agnostic encoder decoder can be trained to have a bottleneck layer that ignores characteristics of specific attacks. By training the encoder-decoder with pairs \mathbf{x}_i and \mathbf{x}_j where i and j represent different types of attacks, the network forgets attack-specific details. Here \mathbf{x}_i and \mathbf{x}_j are the embeddings which are extracted from the w2v2 embeddings passed through penultimate layers of AASIST (denoted henceforth as w2v2-AASIST embeddings.) These embeddings are from two speech utterances for which other attributes such as speaker information and linguistic content may or may not be the same. In our experiments, the AIED was trained with a subset of the ASVspoof 2019 training partition, with mean square error loss,

$$\mathcal{L} = \frac{1}{N} \sum_{\mathcal{D}} \|\tilde{\mathbf{x}}_i - \mathbf{x}_j\|^2, \quad (1)$$

where, $\tilde{\mathbf{x}}_i$ represents the reconstructed output obtained from decoder component of AIED and N represents the total number of training examples.

3.2. Attack generalization using low-rank decomposition

Various domain generalization methods have been proposed in the literature for reducing overfitting of a model to data seen

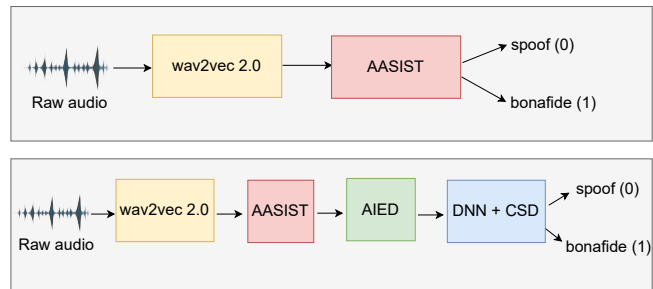


Figure 1: Baseline system (top) and proposed system (bottom).

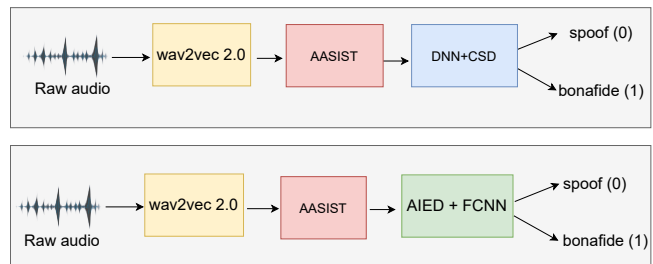


Figure 2: Ablation: with only DNN+CSD (top), with only AIED (bottom).

only during training. The recently proposed Common Specific Decomposition (CSD) [25] technique successfully disentangles common and domain specific components, and demonstrates good generalization to new domains which are not seen during training. CSD works by assuming orthogonality between domain-specific classifiers \mathbf{W}_j and a common classifier \mathbf{W}_c , and by incorporating appropriate loss terms so as to train both \mathbf{W}_j and \mathbf{W}_c . Here j varies from 1 to D_s , which is the number of training domains.

In the context of detecting fake speech, the various attacks present in training data represent the various domains. CSD returns the common classifier \mathbf{W}_c which is attack-agnostic. CSD works in conjunction with the deep neural network denoted by \mathcal{D}_ϕ as a replacement of the final classification layer. This makes the incorporation simple. Following [25], consider the case in which a speech embedding \mathbf{x} is generated as per the following generative model

$$\mathbf{x} = y(\mathbf{x}_c + \sum_{i=1}^d \alpha_{j,i} \mathbf{x}_{s,i}) + \mathcal{N}(0, \Sigma_j) \in \mathbb{R}^m \quad (2)$$

Here the common component \mathbf{x}_c and the individual domain-specific components \mathbf{x}_s are assumed to be orthogonal to each other. The domain-specific components, lying in a latent space of dimension d is correlated with the label y , given by the coefficients $\alpha_{j,i}$. Σ_j represents the attack-specific covariance matrix, capturing attack-specific characteristics.

CSD assumes that the data following the structure outlined in equation 2 can be classified using attack-specific linear classifiers \mathbf{W}_j represented as:

$$\mathbf{W}_j = \mathbf{W}_c + \mathcal{W}_s \gamma_j \quad (3)$$

where \mathbf{W}_j , $\mathbf{W}_c \in \mathbb{R}^{C \times n}$ represent the attack-specific and common classifier respectively. The components of the attack subspace are denoted by $\mathcal{W}_s \in \mathbb{R}^{C \times n \times d}$, where C represents

the number of classes, n is the dimension of the output from the penultimate layer of the neural network \mathcal{D}_ϕ , and d is the latent dimension of the domain space.

Algorithm 1 Common specific decomposition (CSD) [25]

Input:

- $\mathbf{X} \in \mathbb{R}^{m \times N}$ (train data, N : total train examples and m : dimension of each example)
- $\mathbf{y} \in \mathbb{R}^N$ (true/fake labels, where $y \in \mathbf{y}$, and y is either 0 or 1)
- $\mathbf{j} \in \mathbb{R}^N$ (attack labels, where $j \in \mathbf{j}$, and j ranges from 1 to D_s)
- d (attack-specific components: hyperparameter)

Output: \mathbf{W}_c (common classifier)

1. Randomly initialize parameters ϕ of DNN \mathcal{D}_ϕ (values drawn from normal distribution)
 2. Randomly initialize CSD parameters: $\mathbf{W}_c, \mathcal{W}_s, \gamma_j$ (values drawn from normal distribution)
 3. $\hat{\mathcal{W}} = [\mathbf{W}_c, \mathcal{W}_s]$ (concatenation)
 4. $\mathcal{L}_{csd} = 0$ // CSD loss
 5. **for** $(\mathbf{x}, y, j) \in (\mathbf{X}, \mathbf{y}, \mathbf{j})$ **do** //for all batches
 6. $\mathbf{W}_j = \mathbf{W}_c + \mathcal{W}_s \gamma_j$
 7. Compute $\mathcal{L}_c = \text{CrossEntropy}(\hat{y}, y; \phi, \mathbf{W}_c)$
 8. Compute $\mathcal{L}_s = \text{CrossEntropy}(\hat{y}, y; \phi, \mathbf{W}_j)$
 9. Compute $\mathcal{L}_o = \sum_{i=1}^C \mathbf{I}_{d+1} - \hat{\mathcal{W}}[i]^T \hat{\mathcal{W}}[i]_F^2$
where \mathbf{I}_{d+1} is the identity matrix with $(d+1)$ dimension.
 10. $\mathcal{L}_{csd} = \mathcal{L}_{csd} + \mathcal{L}_c + \mathcal{L}_s + \mathcal{L}_o$ //Minimize using SGD
 11. **end for**
 12. Return \mathbf{W}_c
-

Both common and attack-specific classifiers are trained using cross-entropy loss, which uses the common and attack-specific parameters of the CSD layer, respectively. Attack-generalization is further enforced through an orthonormal loss, which promotes orthogonality between the common and attack-specific parameters. This is achieved by minimizing the Frobenius norm between an identity matrix and $\hat{\mathcal{W}}[i]^T \hat{\mathcal{W}}[i]$ which contains CSD layer parameters for i^{th} class, where $\hat{\mathcal{W}} \in \mathbb{R}^{C \times n \times (d+1)}$ is obtained by concatenating \mathbf{W}_c with \mathcal{W}_s . The overall loss function \mathcal{L}_{csd} is the sum of the three loss functions namely: common loss (\mathcal{L}_c), attack-specific loss (\mathcal{L}_s) and orthonormal loss (\mathcal{L}_o).

For testing, the common classifier \mathbf{W}_c learned using CSD, devoid of attack-specific components, is utilized for classification as follows:

$$\hat{y} = \arg \max_i (\text{softmax}(\mathbf{W}_c \mathbf{z})) \quad (4)$$

where \mathbf{z} is the representation from the penultimate layer of \mathcal{D}_ϕ , \hat{y} is the predicted speech utterance label and i represents the class index. This pipeline is henceforth denoted as DNN+CSD.

4. Experiments and Results

The proposed attack-agnostic methods using AIED and CSD are evaluated by enhancing the AASIST pipeline with AIED and CSD, as shown in Figure 1 (bottom panel). Ablation studies, where each of the AIED and CSD modules are removed, are also performed (see Figure 2). The evaluation is done on two

datasets: (a) the ASVspoof 2021 logical access (LA) dataset and deepfake (DF) datasets, and (b) a multilingual Indic synthetic speech dataset. All models studied in this paper use only English data for training, hence Indic language data is completely unseen. This also makes it easier to also compare with other methods.

4.1. Dataset Description

For training, we use the ASVspoof 2019 LA database, which has six types of attacks in the training partition [7]. The first evaluation is on the ASVspoof 2021 LA and DF datasets, which has generated speech passed through various codecs and various compression techniques respectively [8]. The second evaluation is on multilingual speech from five Indian languages, Tamil, Marathi, Gujarati, Hindi and Indian English. The fake speech is generated using the MahaTTS² text-to-speech system, which generates speech in various Indian languages. These constitute the fake speech. For bonafide speech, we use speech from Open-Speech-EkStep dataset [26], which consists of read speech from the same five languages.

4.2. Implementation details

For the baseline model referenced from [3], w2v2 was kept frozen, and AASIST was trained on ASVspoof 2019 LA. All w2v2-AASIST embeddings used in our experiments were extracted from the fifth layer of the wav2vec 2.0 large (XLS-R) model, which incidentally, is trained on multilingual speech. The AIED was trained independently using the w2v2-AASIST embeddings. The AIED consists of an encoder and a decoder, with the encoder comprising two fully connected layers of dimensions 160 and 128, while the decoder mirrors the encoder. Training was conducted for up to 100 epochs using the Adam optimizer with a learning rate of $1e-6$. Once trained, the AIED was frozen for use in the experiments presented in Table 1. The data pairs used to train AIED consist of 10000 speaker- and content-independent, attack mis-matched pairs which were randomly generated from metadata of ASVspoof 2019 [7].

The CSD component was trained using a deep neural network (DNN) architecture. The DNN consists of three fully connected layers with dimensions 128, 64, and 32, respectively. A batch size of 64 and a learning rate of 0.0001 were used with the SGD optimizer.

ASVspoof 2019 development data was used to validate all models presented in Table 1. This was also used to choose the optimal number of attack-specific components d in CSD, which is set to 4.

4.3. Processing pipelines and results

Our baseline system (Figure 1 top panel) is based on [3], and uses a frozen, pre-trained w2v2 front-end followed by the graph neural network based AASIST backend, using the implementation available here³. Embeddings are extracted from the penultimate layer of the AASIST. The proposed system supplements this pipeline using the attack-invariant autoencoder (AIED) and common subspace decomposition using a DNN. For all systems, RawBoost [27] was used for data augmentation.

As tabulated in Table 1, the proposed system gives a relative improvement in EER by 4% for the ASVspoof LA data,

²<https://github.com/dubverse-ai/MahaTTS>

³https://huggingface.co/docs/transformers/en/model_doc/wav2vec2

and by 12% for the ASVspooF DF data⁴. The improvement is more pronounced for the Indic speech data, where the proposed method shows a 34% relative improvement. As can be seen, the results leave much to be desired for the Indic speech data.

We also perform simple ablation studies which use only one of either AIED or DNN+CSD for processing w2v2-AASIST embeddings. The results, tabulated in Table 1 seem to indicate that in many situations, some form of explicit attack-invariance is useful. CSD seems to handle variations within the LA data, whereas AIED seems to be effective for DF data. Whereas for the Indic TTS data, the advantage offered by either AIED or CSD seems to be similar (there is negligible change in results due to the removal of either.)

Table 1: Results in terms of EER (\downarrow) and t-DCF (\downarrow) on the proposed Attack-Agnostic systems on ASVspooF 2021 LA & DF

System	2021 LA		2021 DF		IndicTTS
	EER(%)	t-DCF	EER(%)	EER(%)	
Baseline	6.14	0.3326	12.33	59.82	
Ours	5.84	0.3345	10.9	39.51	
Ablation					
Ours w/o AIED	6.3	0.3558	10.98	39.51	
Ours w/o CSD	5.99	0.337	11.31	39.35	

Table 2: Results in terms of EER(\downarrow) on the proposed Attack-Agnostic systems on IndicTTS dataset

System	IndicTTS	IndicTTS
	Multilingual	English
EER		
Baseline	59.82	44.69
Proposed	39.51	35.73

4.4. Discussion

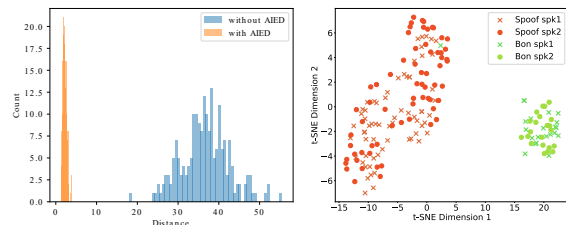
We analyse the improvements brought about by AIED by examining various plots of embeddings before and after AIED. The TSNe plots in Figure 3b suggest that embeddings after AIED are language invariant. The w2v2-AASIST embeddings from 100 examples of fake speech from two languages form distinct clusters, whereas they overlap after AIED. In Figure 3a, TSNe plots indicate that AIED embeddings are invariant to speakers for both real and fake speech. Figure 3a also shows the histograms of distance between 200 pairs of embeddings generated from fake speech using various types of codecs (blue histogram). The figure also shows the same 200 embedding pairs after passing through the AIED network (orange histogram.) The spread of the distance has significantly reduced, indicating that the AIED effectively provides codec invariance.

Similarly, the plots in Figure 3c show the embeddings from w2v2-AASIST and the same embeddings after CSD for two different types of attacks. Before CSD, the two attacks show considerable separation, indicating that w2v2-AASIST embeddings capture attack information. After CSD on the same embeddings, the attacks overlap.

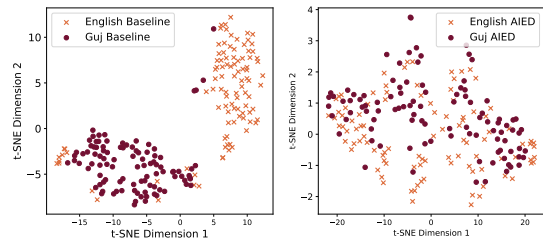
To check the impact of the language seen during training (English) on the models, we additionally evaluate performance

⁴Admittedly, these results are not the state-of-the-art for the ASVspooF 2021 dataset, which may be obtained by fine-tuning the w2v2 model with ASVspooF 2019 data. Limitations in our computing resources prevent us from working with this approach.

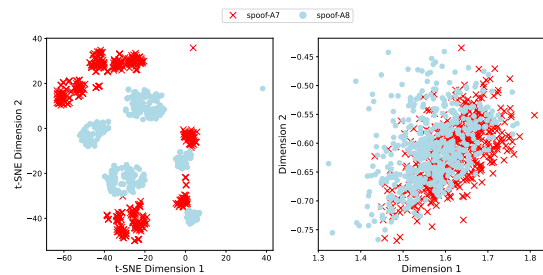
only on Indian English. The improved results shown in Table 2 seem to indicate that language does play a role in the performance obtained. All models were trained on ASVspooF 2019 data, which has only English. Additionally, language identification (LID) experiments on the embeddings after AIED across five Indian languages show significantly poorer LID performance compared to the w2v2-AASIST embeddings, supporting that the proposed embeddings are more language-agnostic than the baseline.



(a) (Left) Histograms of pairwise distance of various codecs before AIED (blue) and after AIED (orange). The use of AIED results in reduced codec spread. (Right) AIED embeddings from two speakers show speaker overlap for both fake and genuine speech.



(b) Plots showing improved language invariance for AIED on two languages from the Indic dataset. (Left) Without AIED, (right) with AIED.



(c) Before (left) and after (right) CSD embeddings.

Figure 3: Visualization of AIED and CSD effects: (a) Codec spread reduction, and improved speaker invariance, (b) improved language invariance, (c) improved attack invariance for CSD embeddings.

5. Conclusions

In this paper, we explored two methods to obtain improved generalization to unseen conditions in the detection of fake speech. Unseen conditions included various transformations applied to speech, as available in the benchmark ASVspooF 2021 dataset, and additionally, speech in unseen languages. The proposed methods show improvements when compared to the well-established w2v2-AASIST pipeline. Although we attempted to provide some insights, future studies will focus on determining the role of various factors which affect generalization to unseen conditions.

6. References

- [1] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.
- [2] 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 transactions on audio, speech, and language processing*, vol. 29, pp. 3451–3460, 2021.
- [3] H. Tak, M. Todisco, X. Wang, J.-w. Jung, J. Yamagishi, and N. Evans, “Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation,” in *The Speaker and Language Recognition Workshop*, 2022.
- [4] J. W. Lee, E. Kim, J. Koo, and K. Lee, “Representation selective self-distillation and wav2vec 2.0 feature exploration for spoof-aware speaker verification,” in *Interspeech 2022*, 2022, pp. 2898–2902.
- [5] H. Wu, J. Zhang, Z. Zhang, W. Zhao, B. Gu, and W. Guo, “Robust spoof speech detection based on multi-scale feature aggregation and dynamic convolution,” in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 10 156–10 160.
- [6] J.-w. Jung, H.-S. Heo, H. Tak, H.-j. Shim, J. S. Chung, B.-J. Lee, H.-J. Yu, and N. Evans, “AASIST: Audio anti-spoofing using integrated spectro-temporal graph attention networks,” in *ICASSP 2022-2022 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2022, pp. 6367–6371.
- [7] M. Todisco, X. Wang, V. Vestman, M. Sahidullah, H. Delgado, A. Nautsch, J. Yamagishi, N. Evans, T. H. Kinnunen, and K. A. Lee, “ASVspoof 2019: Future horizons in spoofed and fake audio detection,” in *Interspeech 2019*, 2019, pp. 1008–1012.
- [8] X. Liu, X. Wang, M. Sahidullah, J. Patino, H. Delgado, T. Kinnunen, M. Todisco, J. Yamagishi, N. Evans, A. Nautsch, and K. A. Lee, “ASVspoof 2021: Towards spoofed and deepfake speech detection in the wild,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 2507–2522, 2023.
- [9] N. Müller, P. Czempin, F. Diekmann, A. Froggyar, and K. Böttinger, “Does audio deepfake detection generalize?” in *Interspeech 2022*, 2022, pp. 2783–2787.
- [10] Bhashini, “Bhashini: National language translation mission,” 2022. [Online]. Available: <https://bhashini.gov.in/>
- [11] H.-s. Shin, J. Heo, J.-h. Kim, C.-y. Lim, W. Kim, and H.-J. Yu, “HM-conformer: A conformer-based audio deepfake detection system with hierarchical pooling and multi-level classification token aggregation methods,” in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 10 581–10 585.
- [12] J. Gong and N. Chen, “Synthetic voice spoofing detection based on feature pyramid conformer,” in *Interspeech 2023*, 2023, pp. 2803–2807.
- [13] H. Tak, J. weon Jung, J. Patino, M. Todisco, and N. Evans, “Graph attention networks for anti-spoofing,” in *Interspeech 2021*, 2021, pp. 2356–2360.
- [14] Y. Guo, H. Huang, X. Chen, H. Zhao, and Y. Wang, “Audio deepfake detection with Self-Supervised WavLM and Multi-Fusion Attentive Classifier,” in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 12 702–12 706.
- [15] X. Wang and J. Yamagishi, “Can large-scale vocoded spoofed data improve speech spoofing countermeasure with a self-supervised front end?” in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 10 311–10 315.
- [16] W. Ge, X. Wang, J. Yamagishi, M. Todisco, and N. Evans, “Spoofing attack augmentation: Can differently-trained attack models improve generalisation?” in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 12 531–12 535.
- [17] L. Zhang, K. A. Lee, L. Zhang, L. Wang, and B. Niu, “CPAUG: Refining copy-paste augmentation for speech anti-spoofing,” in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 10 996–11 000.
- [18] X. Qi, H. Gu, J. Yi, J. Tao, Y. Ren, J. He, and S. Zeng, “MADD: A multi-lingual multi-speaker audio deepfake detection dataset,” in *2024 IEEE 14th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, 2024, pp. 466–470.
- [19] R. Ranjan, B. Dutta, M. Vatsa, and R. Singh, “Faking Fluent: Unveiling the Achilles’ Heel of Multilingual Deepfake Detection,” in *2024 IEEE International Joint Conference on Biometrics (IJCB)*, 2024, pp. 1–10.
- [20] N. M. Müller, P. Kawa, W. H. Choong, E. Casanova, E. Gölge, T. Müller, P. Syga, P. Sperl, and K. Böttinger, “MLAAD: The multi-language audio anti-spoofing dataset,” in *2024 International Joint Conference on Neural Networks (IJCNN)*, 2024, pp. 1–7.
- [21] L. Pepino, P. Riera, and L. Ferrer, “Emotion recognition from speech using wav2vec 2.0 embeddings,” in *Interspeech 2021*, 2021, pp. 3400–3404.
- [22] A. Pasad, J.-C. Chou, and K. Livescu, “Layer-wise analysis of a self-supervised speech representation model,” in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2021, pp. 914–921.
- [23] A. Babu, C. Wang, A. Tjandra, K. Lakhotia, Q. Xu, N. Goyal, K. Singh, P. von Platen, Y. Saraf, J. Pino, A. Baevski, A. Conneau, and M. Auli, “XLS-R: Self-supervised cross-lingual speech representation learning at scale,” in *Interspeech 2022*, 2022, pp. 2278–2282.
- [24] X. Lu, S. Matsuda, C. Hori, and H. Kashioka, “Speech restoration based on deep learning autoencoder with layer-wised pretraining,” in *Interspeech 2012*, 2012, pp. 1504–1507.
- [25] V. Piratla, P. Netrapalli, and S. Sarawagi, “Efficient domain generalization via common-specific low-rank decomposition,” *International Conference on Machine Learning (ICML)*, pp. 7728–7738, 2020.
- [26] Open-Speech-EkStep, “Open-Speech-EkStep Dataset,” 2022. (Accessed: 23 Nov 2022). [Online]. Available: <https://github.com/Open-Speech-EkStep>
- [27] H. Tak, M. Kamble, J. Patino, M. Todisco, and N. Evans, “Rawboost: A raw data boosting and augmentation method applied to automatic speaker verification anti-spoofing,” in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 6382–6386.