Universiteit Leiden
The Netherlands

Opleiding Informatica

Audio Deepfake Detection using HuBERT features

Amy Setropawiro

Supervisors:
Dr. Erwin M. Bakker & Prof. Dr. M. S. Lew

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

10/08/2025

Abstract

There has been a surge of new Audio Deepfake Detection (ADD) techniques in recent years, but most of these techniques lack generalization capabilities and perform poorly on data outside of their training set. This is why recent studies have been more focused on finding features that perform well across different domains. A recent proposed technique tries to achieve this by using a pre-trained Whisper encoder as a feature extractor. In this paper, we propose to use HuBERT as a front-end instead of Whisper in two audio deepfake detection models, SpecRNet and MesoNet. After conducting several experiments, it is shown that the SpecRNet model in combination with HuBERT features is able to reduce the EER to 0.2474, which is lower than the reported EERs of both original baseline techniques using Whisper features.

Contents

1 Introduction

Audio Deepfakes (ADs) are human voices that are manipulated or created using Artificial Intelligence (AI)-tools[2]. There are three main types of these fake generated audios, namely: synthetic-based or Text-To-Speech (TTS), imitation-based or Voice-Conversion (VC), and replay-based. Most studies and datasets focus on TTS and VC deepfakes, since these ADs have been developing at a rapid rate[30]. Although these ADs may also have their advantages such as creating audiobooks and enabling personalized voice assistance, they contribute to a growing cybersecurity concern since people can use these audios to spread misinformation to manipulate the public opinion for propaganda, slander and also terrorism[2]. For example: in 2019, scammers used fake speech audio to mimic a CEO over the phone and were able to obtain a huge amount of money this way[25]. This is why we need to continue to improve and develop new techniques to detect these Audio Deepfakes.

In the last few years, Audio Deepfake Detection (ADD) techniques have mainly been focused on intra-domain conditions and have achieved promising results in this area[31]. However, the problem with these techniques is that they do not perform well when applied to cross-domain scenarios[18]. This is mainly because these techniques do not take the unknown domain (for example audio with background noise) and damaged audio into account[32]. That is why recent research has been more focused on generalization of ADD through feature extraction.

Kawa et al.[12] addressed generalization issues through feature extraction by using Whisper features. Whisper is an automatic speech recognition (ASR) system that is trained on a large magnitude of diverse data. The researchers used this feature extractor in combination with four ADD models (LCNN, MesoNet, SpecRNet and RawNet3) and found that concatenating the Whisper and Mel-frequency cepstral coefficients (MFCC) features for the MesoNet model reduced the equal error rate (EER) to as low as 0.2672 on the In-The-Wild dataset.

Kawa et al. also proposed the SpecRNet model[11], which is a neural network architecture inspired by the RawNet2 backbone. They compared their model to two state-of-the-art architectures(LCNN and RawNet2) and found that the SpecRNet model provided an inference speed improvement of 40%. This in comparison to the LCNN model and while achieving similar ADD results. Kawa et al. also applied Whisper features to this model and were able to lower the EER to 0.3644 when evaluating on the In-The-Wild dataset.

The focus of this thesis will be on trying to adapt ADD techniques such that robustness with respect to domain shifts can be improved. We propose to use and adapt two baseline methods proposed by Kawa et al.[12] which are trained on the ASVspoof2021DF dataset and evaluated on the In-The-Wild dataset. We alter the front-end by using the Hidden-Unit BERT (HuBERT)[7] instead of Whisper as feature extractor and compare the found results to the baseline techniques.

The remainder of the paper is organized as follows: Section 2 contains further information on (audio)deepfake detection techniques and datasets that are related to our approach. In Section 3 the relevant definitions of the used features and performance measure are explained. The two baseline methods that are used to compare our results are discussed in Section 4. Our proposed implementations are defined in Section 5. In Section 6, the experimental setup, datasets, pre-processing, and the train-test split are explained. The discussion of the results we obtained is presented in Section 7. Finally, in Section 8 we discuss our conclusions and further research possibilities.

2 Related Work

In this section, the important benchmark datasets, related and recent (audio)deepfake detection techniques in general, and the state-of-the-art and baseline ADD methods are described.

Several recent ADD techniques try to enhance generalization by improving feature extraction and back-end model classification, but perform poorly when faced with unseen attack types. That is why Huang et al.[8] introduced a novel strategy using Latent Space Refinement(LSR) and Latent Space Augmentation(LSA) to improve the generalization ability of ADD models and were able to surpass several state-of-the-art techniques on the datasets, making this approach a state-of-the-art technique itself. The strategy was evaluated on four different datasets: ASVspoof2019 LA, ASVspoof2021 LA, ASVspoof2021 DF, and In-The-Wild and uses an AASIST model with Wav2Vec2 features. The technique with both LSR and LSA applied outperforms the two baseline techniques of Kawa et al. with an EER as low as 0.0554 on the In-The-Wild dataset.

ADD Benchmark datasets:
There are various datasets used for ADD such as the WaveFake dataset[5], which is a bilingual dataset containing Japanese and English samples and more than 117,000 audio samples that are generated using various TTS systems. It was designed to close the gap between the ADD and the image deepfake detection domain, as the image domain received more attention. Another dataset is the Fake or Real (FoR) dataset[23], which includes more than 198,000 samples from new TTS methods and human speech and is publicly available. It was published in four versions: for-original, for-norm, for-2sec, and for-rerec, and it was created specifically for synthetic voice detection experiments. More popular datasets are the FakeAVCeleb dataset[6] and the ASVspoof2019 dataset[19]. The FakeAVCeleb dataset is a multi-modal deepfake dataset that contains real youtube videos of celebrities from four different racial backgrounds (English speakers) and was designed to be gender and racially unbiased. The samples are generated using an SV2TTS tool and the dataset is widely used not only in ADD but also in video deepfake detection. The ASVspoof2019 dataset was designed for the anti-spoofing challenge for automatic speaker verification(ASV) systems. It consists of two parts, namely: the Logical Access(LA) part which is generated using TTS and VC systems and the Physical Acces(PA) part which contains replay attacks.
The two datasets used by Kawa et al. to train the baseline methods as mentioned above are the following: ASVspoof2021DF[16] and In-The-Wild[18].

- ASVspoof2021DF: This dataset is part of the ASVspoof2021 dataset, which is split into three parts: one for logical access(LA), one for physical access(PA), and one for speech deepfake(DF). The speech deepfake part was not developed to test the robustness of ASV systems like the ASVspoof2019 dataset, but instead to detect fake speech in non-ASV scenarios. Data for the DF part are generated by the TTS and VC algorithms, and the evaluation part contains data from the ASVspoof2019LA dataset[33]. The DF part contains 611,830 audio samples , which includes 1,066 bonafide samples.

- In-The-Wild: This dataset was created by Muller et al.[18] to evaluate their models on realistic data and consists of 37.9 hours of either bonafide(20.7 hours) or fake audio(17.2 hours). The audio samples, which are about 4.3 seconds long, consist of English-speaking voices from politicians and celebrities. The samples are also accumulated from public online platforms.

This dataset is used in various ADD papers to evaluate the cross-domain performance of a model on realistic audio samples.

These two datasets were chosen because the ASVspoof2021DF dataset is one of the largest and most popular DF datasets and the In-The-Wild dataset contains samples which simulate realistic real-world scenarios, for example, samples with background noise.

(Audio)Deepfake Detection techniques:
ADD techniques can be divided into two main groups, namely: Machine Learning(ML) methods and Deep Learning(DL) methods. However, DL classifiers are now being used primarily in the ADD area. Lavrentyeva et al.[13] implemented Light CNN(LCNN) to reduce the model size while also applying the model to the ADD problem on the ASVspoof2019 dataset and achieved an EER of 1.86. Chen et al.[15] presented a channel-wise gated Res2Net(CG-Res2Net), which is a modification of the Res2Net architecture, to enhance the generalization ability of the model on unseen attacks. The model was able to achieve an EER reduction of 28.8% and outperformed Res2Net.
Kawa et al.[11] compared their architecture(SpecRNet) to two different neural networks: LCNN and RawNet2. LCNN has been one of the better performing neural networks in ADD. Wang X. & Yamagishi J.[28] used a loss function based on PS2Grad and combined this with a LFCC front-end and LCNN-LSTM back-end. They ended up achieving an EER of 1.92, which is one of the lowest without data augmentation on the ASVspoof2019 dataset. Tak et al.[26] modified the original RawNet2 architecture to be able to apply it to anti-spoofing. With this architecture, they achieved the second-best results for the A17 attack(a VC-based attack) of the ASVspoof2019 dataset.
When comparing SpecRNet to the two baseline architectures, Kawa et al. found that even though their model had a decrease in computational requirements, it still achieved comparable performance to the other two models. The SpecRNet model could also be easier for the average person to use on their own devices.
A year later, Kawa et al.[12] combined Whisper as front-end with SpecRNet, LCNN, MesoNet and RawNet3. MesoNet has been used a lot in image/video deepfake detection. Afchar et al.[1] used MesoNet(Meso-4 and MesoInception-4) to detect two recent video deepfake techniques(Deepfake and Face2Face). This network consists of alternating layers of convolutions and pooling which are used for extracting features. It also contains a dense network for classification. They achieved a detection rate of around 90% with this architecture. RawNet3 is a novel speaker recognition model, which analyzes raw audio, proposed by Jung et al.[10] The architecture achieved an EER of 0.89 in a supervised learning environment. Compared to the RawNet2 architecture, the EER was reduced from 2.48 to 1.05.

Self-supervised front-ends ADD: X. Wang and J. Yamagishi[29] investigated pre-trained self-supervised front-ends, Wav2Vec 2.0 and HuBERT, in combination with several back-end models on the ASVspoof2019 LA dataset. They compared the results with the baseline using an LFCC-based front-end and found that HuBERT and Wav2Vec 2.0 outperformed those features significantly. They also found that the two fine-tuned front-ends were more robust across ASVspoof2015, ASVspoof2021 LA and DF datasets.
In their research, Li et al.[14] used a HuBERT front-end in combination with a RawNet2 back-end and applied data augmentation for feature extraction and α-FMS to improve the RawNet2 model. They achieved an EER of 2,89 on the ASVspoof2021 LA dataset and their model outperformed the

LFCC front-ends and RawNet2 baselines from the ASVspoof2021 competition.
Combei et al.[3] benchmarked ten pre-trained model representations from three classes of models (self-supervised, speaker embedding, learnable frontends), including HuBERT, Wav2Vec 2.0 and wavLM. They fine-tuned the Wav2Vec2 and wavLM models and applied data augmentation to the ASV5 challenge dataset. Later in their experiment they also applied late fusion to the fine-tuned wavLM models. They found that the self-supervised models outperformed the other two classes, with wavLM achieving an EER of 6.56 on the ASVspoof5 dataset.
Our proposed adapted technique also applies the HuBERT model as a front-end to a back-end model, but in combination with two different back-end models, SpecRNet and MesoNet. To our knowledge, this approach is novel.

3 Fundamentals

In this section, the relevant fundamentals used in this thesis are briefly explained. First, mel-frequency cepstral coefficients(MFCCs), log-Mel spectrograms, and the Whisper and HuBERT front-ends used to extract the features for the (baseline)models will be discussed. Lastly, the evaluation metrics used to measure and compare the performance of the models are explained.

3.1 Mel-Frequency Cepstral Coefficients

The feature representations used by the spectogram-based baseline models (SpecRNet and MesoNet) are mel-frequency cepstral coefficients(MFCC)[9] in concatenation with Whisper features. These MFCCs are important for having distinguishable features that can be used by the networks to differentiate one audio from another.
MFCCs are used a lot in the speech recognition domain[17] because they closely resemble the human hearing system and have been proven to be very effective. Like the name already suggests, MFCCs are coefficients that represent the spectral properties of a sound wave. To get these coefficients, the raw audio is cut into frames, then a Fourier transform is applied to get the frequencies from each frame and lastly these frequencies are divided in mel-scale[9] chuncks. The mel-scale ensures that the frequencies are mapped in a way that equal distances in pitch sound as distant from each other as humans would perceive it.

3.2 Log-Mel spectrograms

The Whisper front-end used by the two baseline models takes log-Mel spectrogram representations as input. A log-Mel spectrogram is a variation of the Mel spectrogram, where a logarithmic transformation is applied to the values of the Mel spectrogram. It has been shown by psychoacoustic experiments[24] that this closely matches the way humans perceive the volume of sound and speech. A Mel spectrogram is a variant of a spectrogram, which converts the frequency scale to the Mel scale. A spectrogram depicts the changes in frequencies over time, and the Mel scale is a perceptual scale that shows distances in pitch that equate to the way humans perceive distances in pitch (see Figure 1).

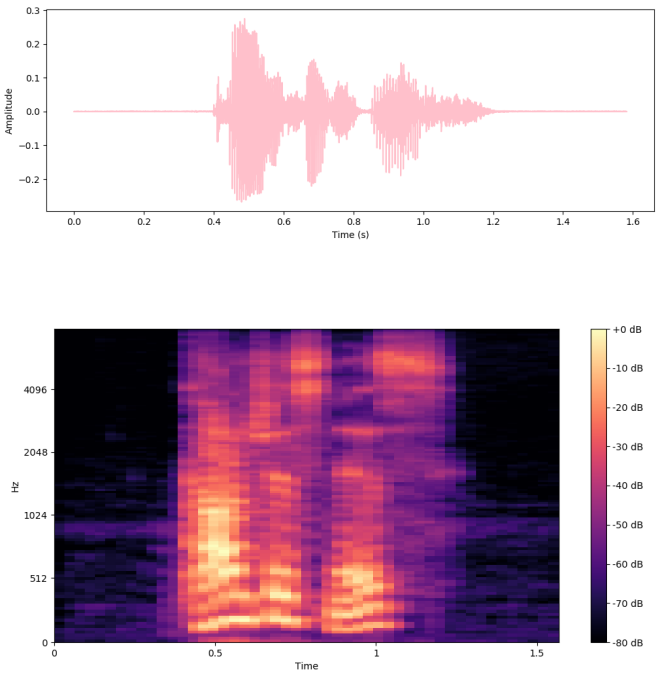


Figure 1: The waveform from a sample of the In-The-Wild dataset with the corresponding generated Mel spectrogram.

3.3 Whisper encoder

As mentioned above, Whisper is a state-of-the-art ASR system[22]. The architecture is trained on 680,000 hours of diverse supervised audio, and because of this diversity is robust against a wide variety of background noise, accents, and languages. In their ADD research, Kawa et al.[12] only used the whisper encoder from the tiny.en variant of the model to extract features for the SpecRNet model. The encoder contains 7,632,384 parameters and outputs data of shape (376, 1500), but to be able to concatenate the front-ends, one of the dimensions is doubled to a tensor of shape (376, 3000). The encoder itself takes 30 seconds of audio converted into a log-Mel spectrogram as input and is based on the encoder-decoder Transformer architecture[27]. It is made up of two convolutional layers, followed by Gaussian Error Linear Unit(GELU) activation functions that process log-Mel spectrograms (See Figure 2). After this, sinusoidal position embeddings are added and lastly the encoder Transformer blocks modify the output. These blocks contain pre-activation residual blocks and a final normalization layer.

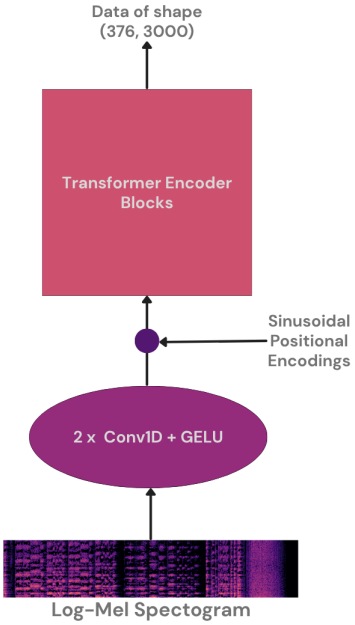


Figure 2: The whisper encoder with log-Mel spectrogram input by Radford et al.[22]

3.4 HuBERT

Hidden-Unit BERT is a self-supervised approach to learning speech representations introduced by Hsu et al.[7]. Here, an offline clustering step is used to generate the target labels of the audio samples. The HuBERT architecture is very similar to Wav2Vec 2.0 and contains a BERT encoder, but the training process is different. BERT was introduced by Devlin et al.[4] and stands for Bidirectional Encoder Representations of Transformers. This model is a Natural Language Processing(NLP) model based on the encoder block of the transformer architecture. The BERT model masks random words (tokens) in from the input text, which it then predicts using surrounding context. Because it is challenging to apply a BERT model to sounds, HuBERT uses clustered hidden units (HU) as pseudo-labels for the self-supervised training (See Figure 3). HuBERT is pre-trained on either Librispeech 960h or Libri-Light 60k hours, and takes raw audio waveform as input. The model used in this research is the HuBERT base model, which contains 95M parameters and is pre-trained on Librispeech 960h. Librispeech is a corpus of read English speech and is obtained from LibriVox audiobooks, a domain of free public audiobooks[20]. Since BERT is designed for text input and HuBERT for audio input, the models are not competing in the same domain.

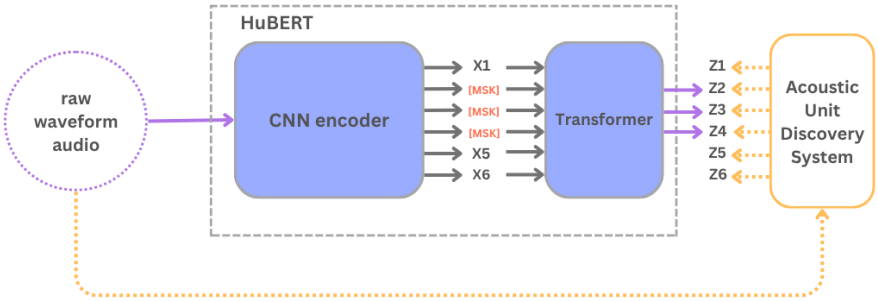


Figure 3: The HuBERT model where some ectracted features(X2, X3 and X4) are masked and an offline clustering unit is used to assign the pseudo-labels(Z2, Z3 and Z4) and predict these.

3.5 Evaluation metrics

To assess the performance of the ADD techniques used in this thesis, the Equal Error Rate(EER), precision, accuracy, and F1-score will be used as metrics. These metrics are widely used in the ADD research field for comparing the ADD performance. We will also use them to compare our techniques with the baseline methods.

3.5.1 Equal Error Rate

The equal error rate(EER) is generally defined as the location on a ROC curve where the false acceptance rate(FAR) is equal to the false rejection rate(FRR)[21]. The lower the EER, the more accurate a technique or model is deemed to be. When the FAR is equal to the FRR, the common value is called the EER. See Equation 1. The false acceptance rate in ADD is the rate at which fake audio samples are incorrectly classified as bonafide. The false rejection rate, on the other hand, is the rate at which bonafide audio samples are incorrectly classified as fake.

$$EER = \frac{FAR + FRR}{2}, where|(FAR - FRR)| = 0 \tag{1}$$

3.5.2 Precision

The precision is defined as the number of True Positives(TP) divided by the number of True Positives and False Positives(FP), which results in Equation 2. The TPs are the fake audio samples correctly identified as fake and the FPs are the bonafide audio samples incorrectly identified as fake.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

3.5.3 Accuracy

The accuracy is defined as the number of True Positives and True Negatives(TN) divided by the total number of samples; the number of True Positives, True Negatives, False Positives and False

Negatives(FN) (see Equation 3). The accuracy in ADD measures the percentage of overall correct predictions made by the model.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

3.5.4 F1-score

The F1-score is generally defined as two times the precision multiplied by the recall, which is then divided by the precision and the recall and results in Equation 5. The F1-score shows the balance between precision and recall. If both precision and recall are high, the F1-score also becomes high. So, the higher the F1-score, the better the performance of the model. Recall is defined in Equation 4 where the number of True Positives is divided by the True Positives and False Negatives. The False Negatives are the fake audio samples incorrectly classified as bonafide.

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

$$F1 - score = \frac{2 \times precision \times recall}{precision + recall} \tag{5}$$

4 Baseline methods

In this Section, the two baseline methods used in this thesis are explained. The first baseline method is the SpecRNet model with Whisper features. This model was proposed and implemented by Kawa et al.[12] and achieved a reported EER of 0.3644 when combined with Whisper features and evaluated on the In-The-Wild dataset. The second baseline method is the MesoNet model with joint Whisper and MFCC features. This technique performed the best and reported an EER as low as 0.2672 when fine-tuned and evaluated on the In-The-Wild dataset. The two architectures were chosen because they achieved the best results of all models compared at the time by Kawa et al. The implementation of these baseline methods is publicly available and is taken from the Github of Kawa et al.: github.com/piotrkawa/deepfake-whisper-features.

4.1 Whisper SpecRNet

The SpecRNet used by Kawa et al. combined with Whisper features differs slightly from the original implementation[11] and contains adaptive 2D average pooling after the last SeLU layer in the pre-recurrent normalization layer to be able to process higher-dimension front-ends. The SpecRNet architecture is inspired by the RawNet2 model and contains 227,963 parameters.
The architecture we use as a baseline model(see Figure 4)is equal to Kawa et al.'s implementation, it starts with the pre-trained Whisper encoder as front-end (see Section 3) and ends with the SpecRNet model as back-end. The SpecRNet architecture starts with a preliminary normalization layer that contains 2D batch normalization and the SeLU activation function. The network then continues with three residual blocks that are all succeeded by an FMS attention layer and ends with a pre-recurrent normalization layer, two bidirectional GRU layers, and two fully connected layers. The residual blocks contain two convolutional layers, a 2D batch normalization layer and a leaky

ReLU function. The last two residual blocks also contain a batch normalization layer and a leaky ReLU function at the start of the block. The FMS attention layers contain a 2D max pooling layer succeeded by an FMS attention block, and lastly a 2D max pooling layer again.

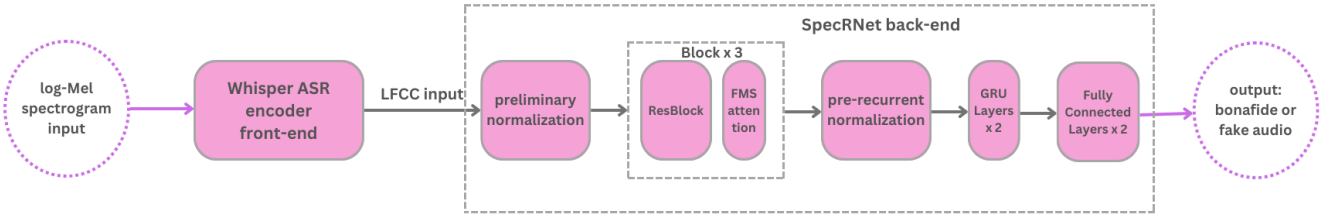


Figure 4: A schematic representation of the Whisper feature extractor and the SpecRNet model used by the baseline technique.

4.2 Whisper + MFCC MesoNet

The MesoNet model used in this architecture (see Figure 5) is the MesoInception-4 variant implemented by Afchar et al.[1] with a small modification(adaptive 1D average pooling) almost at the end in front of the first fully connected layer and contains 28,486 parameters.
The model was chosen as a baseline because it was the best performing model using the concatenated front-end features compared by Kawa et al. This network starts off with a concatenation of the Whisper and MFCC features as input. It then continues with the MesoNet architecture which contains an inception module, two layers of consecutive convolutions and pooling, and lastly two fully-connected layers (which use dropout).
The inception module contains several stacked convolutional layers with non-identical kernel shapes as well as 3×3 dilated convolutions to handle multi-scale information. The two layers of convolutions use a RelU activation function and batch normalization to improve generalization abilities. Again, to be able to process the higher-dimension front-ends, this architecture contains 1D average pooling in front of the first fully connected layer.

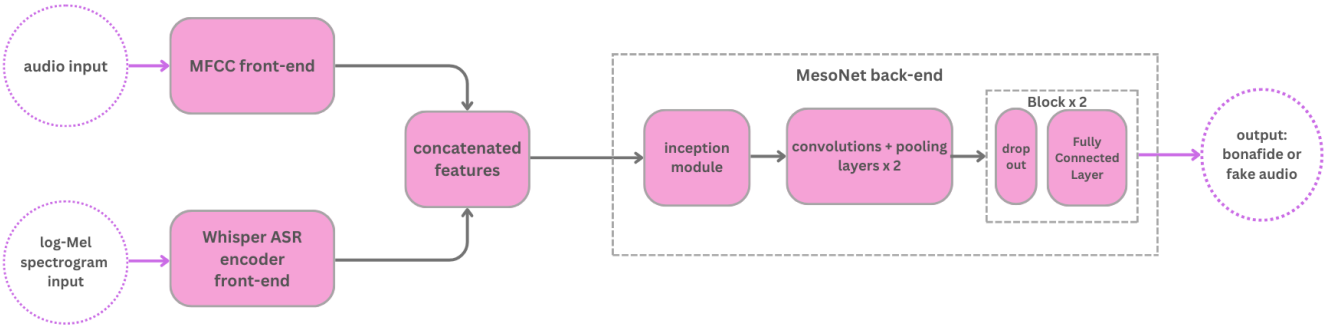


Figure 5: A schematic representation of the concatenated MFCC and Whisper features and the MesoInception-4 model used by the baseline technique.

5 Proposed technique: HuBERT features

For our implementation, we will use the pre-trained HuBERT model[7] to extract HuBERT features to use on the baseline models instead of Whisper features and compare our results to the baseline methods. By doing this, we will try to improve the generalization abilities of the techniques. We will also train and test our technique on the ASVspoof2021 DF[16] dataset and evaluate our technique on the In-The-Wild[18] dataset in the same way as the original two baseline methods have been evaluated.

We opt to use HuBERT as the front-end because HuBERT has been showing promising results in performance and generalization. This encoder model is able to extract robust features that can be applied to an unknown domain without being trained on labeled datasets. The HuBERT architecture we use to extract the HUBERT features is the feature encoder that exists of a convolutional waveform encoder and a BERT encoder, as can be seen in Figure 6. Since we use HuBERT as the feature extractor in our implementation and do not pre-train the HuBERT model, the weights remain frozen. This means that the architecture does not mask input tokens to predict masked targets, as was done during the training of the HuBERT architecture by Hsu et al.[7]. It also does not perform k-means clustering on MFCCs to generate discrete labels as targets, as was also done during training of the HuBERT model by Hsu et al.

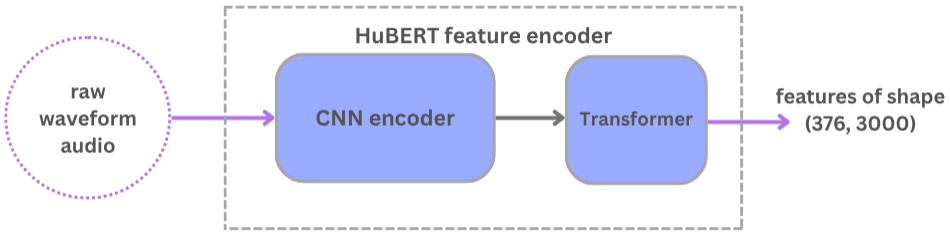


Figure 6: The HuBERT feature encoder as used by our proposed technique.

We combine the HuBERT front-end with the two best performing back-ends of the baseline models,

SpecRNet and MesoNet. Kawa et al.[12] found that these two back-ends were able to generalize better and provided greater efficacy than the LCNN model. The architecture of the models with the HuBERT front-end can be seen in Figure 7 and Figure 8. For the first implementation, we applied the HuBERT features to the SpecRNet model without changing anything about the SpecRNet architecture used by the baseline model. The same is done for the MesoNet model, where we also do not alter anything in the architecture and only concatenate the HuBERT features with the MFCC features.

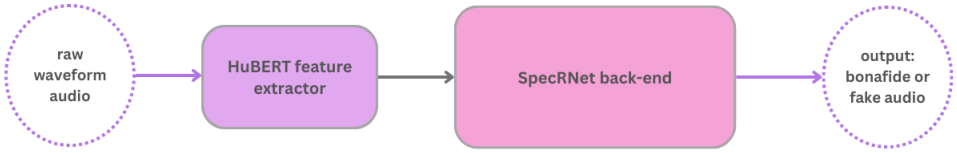


Figure 7: A simple schematic representation of the HuBERT feature extractor and SpecRNet model.

The related research in Section 2 on self-supervised front-ends in ADD shows that HuBERT has been used as front-end in combination with several back-end models in several research papers and has also been able to obtain good results. Since HuBERT has not been used as a front-end with the SpecRNet and MesoNet models as back-end and in concatenation with other features, the two baseline models are altered to apply HuBERT features.

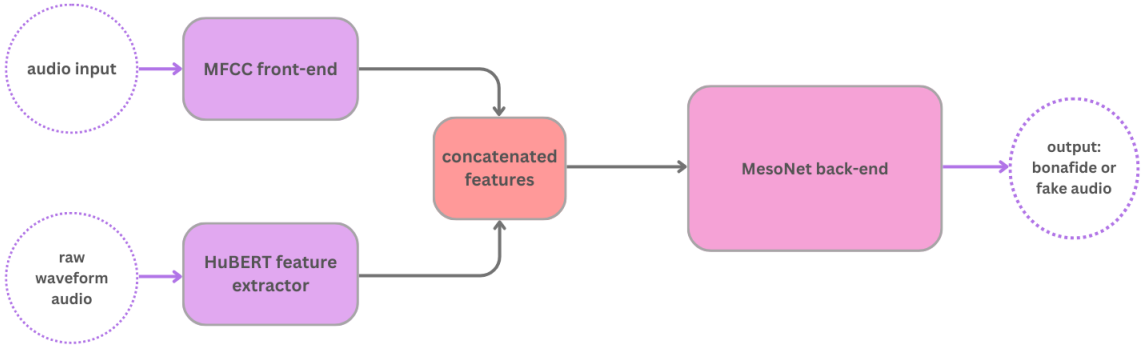


Figure 8: A simple schematic representation of the HuBERT and MFCC concatenated features and the MesoNet model.

6 Experimental Setup

In this section, the datasets ASVSpoof2021 DF and In-The-Wild, the preprocessing, and the setup used for training and evaluation of our proposed techniques will be discussed. We will also explain

the train-test split used to execute the experiments. For the setup, we used the code from Kawa et al. that was already available and only altered it slightly to implement HuBERT by Hsu et al. as front-end.

6.1 Datasets

The datasets, ASVspoof2021 DF and In-The-Wild, used by Kawa et al. to train and evaluate the baseline models are the same datasets used to train and evaluate our proposed techniques. In Section 2, general information about the two datasets is discussed. The ASVspoof2021 DF dataset is chosen by Kawa et al. because DF datasets are generally scarce and this DF dataset is one of the largest and most used dataset in ADD research. The In-The-Wild dataset is chosen because it contains samples that reflect real-world scenarios, and by using this dataset, the generalization abilities of the models can be tested.
To train our techniques, we use a random subset of the ASVspoof2021 DF dataset, and to balance the disproportion between the fake and bonafide samples in this dataset, we use oversampling. Kawa et al. used a random subset because they wanted to make their solution trainable on one GPU in a time span of 24 hours, and since they used a tiny version of Whisper, they did not expect a gain from a larger dataset. Since we also used the smallest HuBERT version (HuBERT Base) in our proposed adapted models, we chose to use a random subset of the ASVspoof2021 DF dataset.

6.2 Preprocessing

For the preprocessing step, each sample from the ASVSpoof2021 DF and In-The-Wild datasets is resampled to 16 kHZ mono-channel, and silences longer than 0.2 s are removed. The samples were also padded (by repetition) and trimmed to 30 s of content by Kawa et al. for two reasons: Whisper has a fixed input limit that is 30 s, which means that the model takes exactly 30 s of audio as input, so instead of padding with zeros the whole tensor is filled with audio, and work from Muller et al. also shows that analyzing longer utterances gives better results. Although Hubert does not have a fixed input limit, we wanted to keep the experimental setup as close to the original as possible to compare the techniques; hence we kept this procedure the same.

6.3 Train-test split

The train-test split used by Kawa et al. and for our setup is 70-15 from the ASVSpoof2021 DF dataset, where 70%(428,279 samples) is used for the training set and 15% (91,775 samples) is used for the validation set, as can be seen in Table 1. The 428,279 samples from the training set consist of 412,448 fake samples and 15831 bonafide samples. The 91,775 validation samples consist of 88,382 fake samples and 3393 bonafide samples. To keep the same train-test split as Kawa et al. [12], we also randomly select 100,000 samples from the 428,281 samples to train our model and 25,000 samples to validate our model. The remaining 15% of the dataset, which is usually left for testing the model, is not used, since all 31,779 samples from the In-The-Wild dataset are used for this purpose.

Dataset Split	Available Samples	Used for Model
Train (ASVspoof2021DF)	428,279	100,000
Validation (ASVspoof2021DF)	91,775	25,000
Test (ASVspoof2021DF)	91,775	0
Total (ASVspoof2021DF)	**611,829**	**125,000**
Test (In-The-Wild)	31,779	31,779

Table 1: Dataset train-test split and usage

6.4 Training

This Section discusses how the models with our proposed front-end were trained. The training setup is similar to the training setup used for the baseline models, and the same hyperparameters as the baselines were used.

HuBERT SpecRNet
To train the SpecRNet model with the HuBERT front-end, a binary cross-entropy function is used for 10 epochs with a batch size of 8. During these 10 epochs, a learning rate of 10^{-4} and a weight decay of 10^{-4} are applied. During training, the weights of HuBERT remain frozen so that SpecRNet is trained with the pre-trained HuBERT features.

HuBERT + MFCC MesoNet
To concatenate the HuBERT front-end with the MFCC front-end, the output data shapes of both front-ends must be matched. The HuBERT output data shape is (768, 1499), which is different from the MFCC output data shape of (384, 3000). To match the tensors, one of the HuBERT dimensions is resized using interpolation to a shape of (768, 3000). One dimension of MFCC is also resized with a convolutional layer to a tensor shape of (768, 3000). After matching the tensors, the MesoNet model with the HuBERT front-end is also trained for 10 epochs and a batch size of 8 with a binary cross-entropy function(using a learning rate of 10^{-4} and a weight decay of 10^{-4}). During the 10 epochs, the HuBERT weights remain frozen again, but are unfrozen for an additional 5 epochs of training for fine-tuning. During the fine-tuning process, only the CNN encoder is unfrozen. The batch size is reduced from 8 to 4 in this step because less GPU memory is consumed this way. This additional training with the unfrozen HuBERT front-end is performed because Kawa et al. found that fine-tuning the Whisper front-end to the DF detection problem, especially for the MesoNet model in concatenation with the MFCC features, provided the best results. Since we want to fairly compare our technique to the baseline method, we chose to also perform fine-tuning on this model.

7 Experimental Results

In this section, the experimental results of our proposed techniques and the baseline methods will be compared and discussed.

7.1 Experimental Results of the Baselines

In this section, the results of the baseline methods obtained by Kawa et al. and the reproduced baselines will be compared and discussed. In Table 2 the EER results of the baselines obtained by Kawa et al. can be seen, as well as the results obtained by running the baselines on our machine. The baseline results obtained by Kawa et al. for the SpecRNet model with Whisper features and the MesoNet model with Whisper + MFCC features are 0.3644 and 0.3822 respectively. The results we obtained for the two baselines are 0.3626 and 0.4846. The difference in EER results of the SpecRNet model with Whisper features is minimal, indicating that the baseline has good reproducibility. However, the original and reproduced MesoNet model with Whisper + MFCC features show a significantly larger difference in EER results. This difference in EER could be related to a difference in implementation, training setup, or even hardware inconsistencies.

Model	Front-end	EER(frozen)	EER(tuned)
SpecRNet[1]	Whisper	0.3644	-
Baseline SpecRNet	Whisper	0.3626	-
HuBERT-SpecRNet	HuBERT	**0.2474**	-
MesoNet[1]	MFCC + Whisper	**0.3822**	**0.2672**
Baseline MesoNet	MFCC + Whisper	0.4846	0.3041
HuBERT-MesoNet	MFCC + HuBERT	0.4119	0.3963

Table 2: The original and reproduced EER results of the baseline models with the Whisper front-end and the EER results of the same models with the HuBERT front-end. [1]As reported by Kawa et al. in [12]. The other results are obtained by training and evaluating the models on our own system using the experimental setup as described in Section 6. The tuned EER column contains the results of the models with a fine-tuned front-end on the ASVspoof2021 DF dataset.

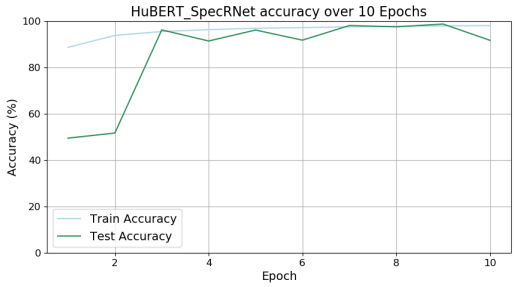
Model	Front-end	Precision	Accuracy (%)	F1-score
Baseline SpecRNet	Whisper	0.8019	58.3774	0.5749
SpecRNet	HuBERT	**0.9140**	58.0249	0.5230
Baseline MesoNet	MFCC + Whisper	0.6568	49.7860	0.5126
MesoNet	MFCC + HuBERT	0.7070	58.7487	0.6410
Baseline MesoNet (tuned)	MFCC + Whisper	0.8380	**67.1010**	0.6927
MesoNet (tuned)	MFCC + HuBERT	0.6626	61.9870	**0.7267**

Table 3: Precision, accuracy, and F1-scores of the two baseline models trained and evaluated on our own system with the Whisper front-end and the proposed HuBERT front-end using the experimental setup as described in Section 6.

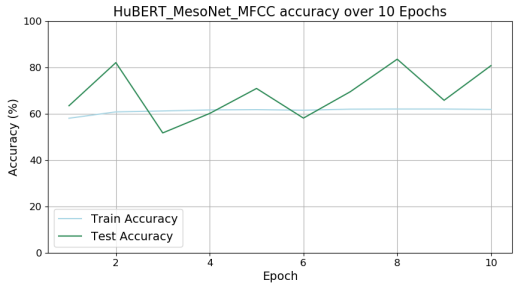
7.2 Experimental Observations: Proposed techniques

In this section, the results obtained with the proposed techniques will be discussed and compared. The results obtained with the proposed SpecRNet model with HuBERT features and the MesoNet model with concatenated HuBERT and MFCC features can be seen in Table 2 and Table 3. The original paper by Kawa et al.[12] did not have the results from Table 3 for the two baseline models,

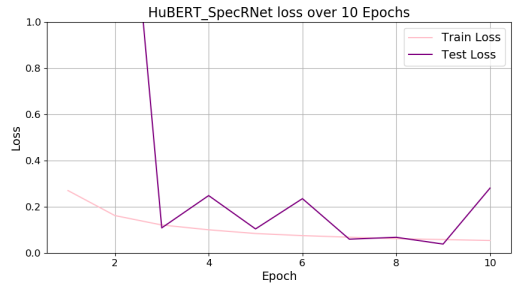
because they chose to compare the models based solely on EER. This could be because EER is a key metric in ADD research and provides a balanced measure. In this case, for example, the dataset is imbalanced, and since the F1-score depends on precision and recall, the score can be skewed by this imbalance.
The SpecRNet model with HuBERT features performed well with an EER of 0.2474. However, the MesoNet model with concatenated HuBERT and MFCC features performed worse than the previous model with an EER of 0.4119.
The precision, accuracy, and F1-score of the two proposed models can be seen in Table 3. Both models have a relatively low accuracy of around 58%. However, the SpecRNet model with HuBERT features has high precision but a low F1-score, while the MesoNet model with concatenated HuBERT and MFCC features has low precision but a higher F1-score. This suggests that the MesoNet model achieves a overall better detection balance. When comparing the results of the fine-tuned MesoNet model with the frozen MesoNet model, an improvement in EER can be observed (from 0.4119 to 0.3963), as well as an improvement in accuracy and F1-score.
The train/test accuracy and loss graphs of the two proposed models can be explored by looking at Figure 9. When comparing the accuracy graphs of the SpecRNet model with the HuBERT model, the SpecRNet model has a higher and more stable test accuracy while the MesoNet has a low and unstable test accuracy. The train/test accuracy and loss graphs of the fine-tuned MesoNet model with concatenated HuBERT and MFCC features can be seen in Figure 10. The accuracy graph reveals that the MesoNet model is likely overfitting and the generalization ability is reducing after the second epoch.



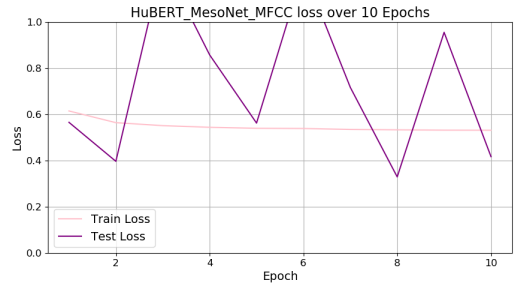
(a) Accuracy graph HuBERT SpecRNet



(b) Accuracy graph HuBERT+MFCC MesoNet



(c) Loss graph HuBERT SpecRNet



(d) Loss graph HuBERT+MFCC MesoNet

Figure 9: The train/test loss and accuracy graphs of the SpecRNet and MesoNet model with HuBERT (and MFCC) features over 10 epochs of training.

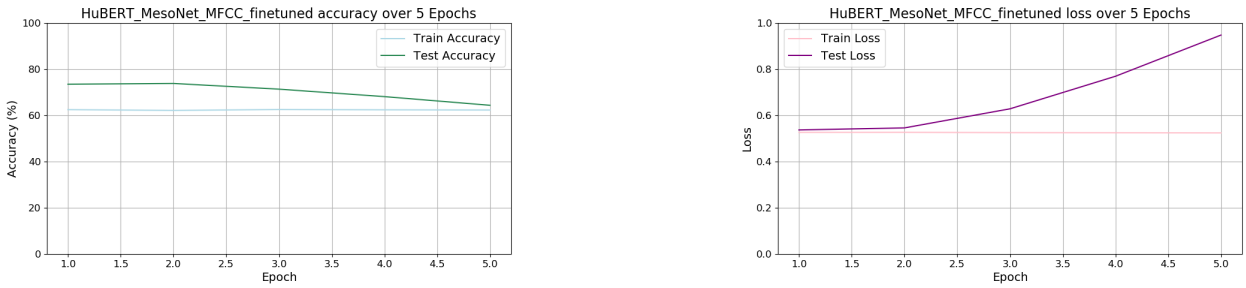


Figure 10: The train/test loss and accuracy graph of the MesoNet model with HuBERT + MFCC features over 5 epochs of finetuning.

The EER results of the baselines, together with the proposed techniques, can be seen in Table 2. The SpecRNet model with HuBERT features has a lower EER than both the original and the reproduced baseline EER. However, the MesoNet model with concatenated HuBERT and MFCC features has a higher EER than the original baseline but a lower EER than the reproduced baseline. After fine-tuning the proposed MesoNet model with concatenated HuBERT and MFCC features, the EER of the original baseline is still lower than the proposed model. The proposed SpecRNet model with HuBERT features has the lowest EER of 0.2474, even lower than the original fine-tuned MesoNet model (0.2672) with concatenated Whisper and MFCC features, which was the best performing model in Kawa et al.'s research.
The precision, accuracy and F1-score in Table 3 show a nearly identical accuracy for the SpecRNet model with Whisper features as well as HuBERT features. The baseline model has a higher F1-score but a lower precision than the HuBERT-based SpecRNet. The metrics of the frozen MesoNet model with concatenated HuBERT and MFCC features show a clear improvement compared to the baseline model. This can not be said about the fine-tuned MesoNet model in comparison to the fine-tuned baseline model, where only the F1-score is slightly higher. This suggests that HuBERT features outperform Whisper features in a frozen setting, but fine-tuning boosts the performance for both models.

7.3 Discussion

From the results discussed in the above subsections, the SpecRNet model with HuBERT features is able to achieve the overall lowest EER and highest precision, indicating that HuBERT features in combination with the SpecRNet back-end are the most effective on the benchmark dataset. This is because compared to the results obtained by Kawa et al. of the SpecRNet model with Whisper features on the benchmark dataset, the SpecRNet model with HuBERT features reduced the EER to 0.2474.
The MesoNet model with HuBERT + MFCC features did not perform as hoped, with an EER of 0.4119, and shows mixed results and unstable performance. This is why the results of the MesoNet model with HuBERT + MFCC features should be interpreted with caution, since there might be potential limitations that may affect their reliability. Due to time constraints, we were not able to look further into this. Although there can be a clear improvement seen in some aspects, the results of the original fine-tuned baseline model of Kawa et al. are still better, with an EER of

0.2672. The HuBERT-based MesoNet model also shows signs of overfitting during fine-tuning, which might suggest that the model is not able to generalize well with the concatenated features. The low accuracy after fine-tuning enforces this suggestion.
Since the EER reflects the balance between the FPs and FNs, it captures the overall balance of the models, and thus is the most important and decisive metric.

8 Conclusions and Future Research

The performance of the two best performing models, SpecRNet and MesoNet, with the Whisper features of Kawa et al.[12] is compared to the same models but with the HuBERT features proposed in this research. This in hopes of being able to find a technique which takes scalability for future challenges into account by focusing on generalization abilities. The SpecRNet model in combination with HuBERT[7] features obtains the best performance with an EER of around 0.25 and shows improved generalization abilities when trained on the ASVspoof2021 DF dataset[16] and evaluated on the In-The-Wild dataset[18].
For further research, a deeper investigation of MesoNet with concatenated HuBERT and MFCC features could be conducted to better understand the instability in performance and the reason why the technique is overfitting, which in turn could be beneficial in finding a technique with potential better generalization abilities.
The main contributions added to the experiments done by Kawa et al. are proposing the HuBERT features in combination with the two best-performing models instead of the Whisper features and a comparison in performance between the proposed techniques and the baseline techniques.

References

[1] D. Afchar et al. "MesoNet: a Compact Facial Video Forgery Detection Network". In: *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. 2018, pp. 1–7.

[2] Z. Almutairi and H. Elgibreen. "A review of modern audio deepfake detection methods: Challenges and future directions". In: *Algorithms* 15.5 (May 2022), p. 155.

[3] D. Combei et al. "WavLM model ensemble for audio deepfake detection". In: *Proc. ASVspoof 2024*. 2024, pp. 170–175.

[4] J. Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.

[5] J. Frank and L. Schönherr. "WaveFake: A Data Set to Facilitate Audio Deepfake Detection". In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Ed. by J. Vanschoren and S. Yeung. Vol. 1. 2021.

[6] K. Hasam et al. "FakeAVCeleb: A Novel Audio-Video Multimodal Deepfake Dataset". In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Ed. by J. Vanschoren and S. Yeung. Vol. 1. 2021.

[7] W. Hsu et al. "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 3451–3460.

[8] W. Huang et al. "Generalizable audio deepfake detection via latent space refinement and augmentation". In: *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2025, pp. 1–5.

[9] A. Jellali, I. Ben Fredj, and K. Ouni. "Pushing the boundaries of deepfake audio detection with a hybrid mfcc and spectral contrast approach". In: *Multimedia Tools and Applications* (July 2024).

[10] J. Jung et al. "Pushing the limits of raw waveform speaker recognition". In: *Interspeech 2022*. 2022, pp. 2228–2232.

[11] P. Kawa, M. Plata, and P. Syga. "SpecRNet: Towards Faster and More Accessible Audio DeepFake Detection". In: *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. 2022, pp. 792–799.

[12] P. Kawa et al. "Improved DeepFake Detection Using Whisper Features". In: *Interspeech 2023*. 2023, pp. 4009–4013.

[13] G. Lavrentyeva et al. "STC Antispoofing Systems for the ASVspoof2019 Challenge". In: *Proc. Interspeech 2019*. 2019, pp. 1033–1037.

[14] L. Li et al. "Voice deepfake detection using the self-supervised pre-training model hubert". In: *Applied Sciences* 13.14 (2023), p. 8488.

[15] X. Li et al. "Channel-Wise Gated Res2Net: Towards Robust Detection of Synthetic Speech Attacks". In: *Interspeech 2021*. 2021, pp. 4314–4318.

[16] X. Liu et al. "ASVspoof 2021: Towards Spoofed and Deepfake Speech Detection in the Wild". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 31 (2023), pp. 2507–2522.

[17] A. Mahmood and U. Köse. "Speech recognition based on convolutional neural networks and MFCC algorithm". In: *Advances in Artificial Intelligence Research* 1.1 (2021), pp. 6–12.

[18] N. Müller et al. "Does Audio Deepfake Detection Generalize?" In: *Interspeech 2022*. 2022, pp. 2783–2787.

[19] A. Nautsch et al. "ASVspoof 2019: Spoofing countermeasures for the detection of synthesized, converted and replayed speech". In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 3.2 (2021), pp. 252–265.

[20] V. Panayotov et al. "Librispeech: an asr corpus based on public domain audio books". In: *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2015, pp. 5206–5210.

[21] I. Pattnaik, A. Dev, and A.K. Mohapatra. "A face recognition taxonomy and review framework towards dimensionality, modality and feature quality". In: *Engineering Applications of Artificial Intelligence* 126 (2023), p. 107056. ISSN: 0952-1976.

[22] A. Radford et al. "Robust Speech Recognition via Large-Scale Weak Supervision". In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by A. Krause et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 23–29 Jul 2023, pp. 28492–28518.

[23] R. Reimao and V. Tzerpos. "For: A dataset for synthetic speech detection". In: *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*. IEEE. 2019, pp. 1–10.

[24] S. S. Stevens and J. Volkmann. "The relation of pitch to frequency: A revised scale". In: *The American Journal of Psychology* 53.3 (1940), pp. 329–353.

[25] C. Stupp. *Fraudsters used AI to mimic CEO's voice in unusual cybercrime case - WSJ*. Aug. 2019. URL: https://www.wsj.com/articles/fraudsters-use-ai-to-mimic-ceos-voice-in-unusual-cybercrime-case-11567157402.

[26] H. Tak et al. "End-to-End anti-spoofing with RawNet2". In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 6369–6373.

[27] A. Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.

[28] X. Wang and J. Yamagishi. "A Comparative Study on Recent Neural Spoofing Countermeasures for Synthetic Speech Detection". In: *Interspeech 2021*. 2021, pp. 4259–4263.

[29] X. Wang and J. Yamagishi. "Investigating Self-Supervised Front Ends for Speech Spoofing Countermeasures". In: *The Speaker and Language Recognition Workshop (Odyssey 2022)*. ISCA. 2022.

[30] Y. Xie et al. "Domain Generalization via Aggregation and Separation for Audio Deepfake Detection". In: *IEEE Transactions on Information Forensics and Security* 19 (2024), pp. 344–358.

[31] Y. Xie et al. "Learning A Self-Supervised Domain-Invariant Feature Representation for Generalized Audio Deepfake Detection". In: *INTERSPEECH 2023*. 2023, pp. 2808–2812.

[32] Y. Xie et al. "Single Domain Generalization for Audio Deepfake Detection". In: *Proceedings of the Workshop on Deepfake Audio Detection and Analysis co-located with 32th International Joint Conference on Artificial Intelligence (IJCAI 2023), Macao, China, August 19, 2023*. Vol. 3597. CEUR Workshop Proceedings. CEUR-WS.org, 2023, pp. 58–63.

[33] J. Yamagishi et al. "ASVspoof 2021: accelerating progress in spoofed and deepfake speech detection". In: *ASVspoof 2021 Workshop - Automatic Speaker Verification and Spoofing Coutermeasures Challenge*. Sept. 2021.