UNMASKING THE FAKE: DETECTION TECHNIQUES FOR DEEPFAKE AUDIO

Mohith Lavu

Abstract—The rapid growth of deepfake audio technology has made it increasingly challenging to differentiate between real and synthetic voices, leading to significant risks such as identity theft, misinformation, and security breaches. To tackle this issue, this project develops a deepfake audio detection system that utilizes multiple models to effectively identify fake audio. The proposed system leverages advanced techniques to analyze various characteristics of audio signals, enabling the detection of subtle discrepancies that are typical of deepfake content. By combining different approaches, the system enhances its ability to distinguish real speech from synthetic speech with improved accuracy and reliability. This research aims to contribute to the development of more effective tools for deepfake audio detection.

1 Introduction

The development of deepfake audio technologies has remarkably changed speech synthesis by making it possible to generate highly realistic synthetic voices. This new technology has advantages in entertainment, virtual assistants, and accessibiltiy solutions, however, it also poses enormous security risks such as identity fraud, misinformation, and social engineering attacks. Perpetrators can take advantage of deepfake audio to assume people's identities, distort information, or bypass voice authentication systems. This necessitates the imposition of dependable methods of detection. In this project, we attempt to devise a system that attempts to determine with high accuracy if an audio sample is real or fake. The goal of this project is to construct an audio deepfake detection system which utilizes three models: a Convolutional Neural Network (CNN), a hybrid with Multi-Layer Perceptron (MLP), and a hybrid with Long Short-Term Memory (LSTM). Accomplishing all mentioned goals requires building a complete audio deepfake detection system which involves designing the entire architecture of the CNN model first, which acts as the baseline. This model will perform feature extraction, which includes gathering spectral and temporal features from audio signals, then classify the subset of audio signals into different categories. For the hybrid model, the classification step is enhanced by the use of MLP which adds greater ability for complex decision boundaries, overcoming interclass overlap issues. The hybrid model was proven to perform better than the simple CNN model because it is better at capturing sequential dependencies embedded within audio signals.

2 Related Work

Stemming from the dangers identity theft, misinformation and breaches in security pose, the detection of deepfake audio remains a critical focus area for research. Numerous investigations have attempted utilizing data augmentation techniques alongside various feature extraction methods in deep learning models aimed at improving the classification of deepfake speech.

The detection of deepfake audio has greatly benefitted from deep feature extraction, especially with ResNet and RawNet2 that exemplify their prowess in using such techniques. In Ä Comprehensive Survey with Critical Analysis for Deepfake Speech Detection,ẗhe analysis offered includes numerous models and their attempts at detecting real and synthetic speech, along with claims that some of the models did not perform up to the set expectations. As for some of the best performing models, Whisper+MLP respectively achieved an accuracy rate of 0.85 along with F1 score of 0.88, AUC of 0.95, and 0.10 error rate, all with no data augmentation by using raw audio. CNN, a notable model that computes features of STFT&LF and STFT&GAM processed audio with Mixup and Spec data augmentation, performed better reporting 0.88 accuracy along with 0.90 F1 score, 0.98 AUC, and 0.05 error rate. Furthermore, a combination of raw audio and STFT&LF features with Mixup and Spec augmentation processed by Whisper+CNN led to an achievement of 0.87 accuracy, 0.89 F1 score, 0.99 AUC, and 0.03 error rate.

While these models certainly work, there is a lack of research focused on CNN-MLP hybrid models tailored to deepfake speech detection. Although CNN-based models extract spatial and spectral features efficiently, classifying their features with MLP may improve classification performance through the capturing of complex boundaries for decisions. This study proposes a CNN-MLP hybrid model and assesses its performance against two other models: a CNN-based standalone model and a hybrid model with LSTM. Through these methods, this study seeks to advance the state of the art in deepfake audio detection systems and contribute to their enduring development.

3 Implementation Process

3.1 Model Selection and Justification

In developing our deepfake speech detection system, we selected Convolutional Neural Networks (CNNs) as the foundational architecture for several compelling reasons:

1. Proven Effectiveness in Deepfake Detection: CNNs have shown marked advantage in the detection of deepfake audio. Research shows that models based on Convolutional Neural Networks (CNN) perform better with lower Equal Error Rates (EER) than Recurrent Neural Networks (RNNs) and even hybrid models. This implies that CNNs specialize in noticing the frequency-based differences which are typical of deepfakes.

2. Expertise and Familiarity: Leveraging prior experience with CNN architectures allows for more efficient model development and optimization. This familiarity facilitates the design of models tailored to the specific nuances of deepfake audio detection.

3. Adaptability to Various Input Features: CNNs are versatile in processing different spectrogram representations, such as Short-Time Fourier Transform (STFT), Constant-Q Transform (CQT), and Wavelet Transform (WT). This flexibility enables the exploration of multiple feature extraction techniques to enhance detection accuracy.

4. Scalability and Efficiency: The hierarchical structure of CNNs allows for efficient learning of both low-level and high-level features, making them scalable to larger datasets and complex patterns inherent in audio signals.

By building upon a CNN foundation, we aim to harness these strengths to develop a robust and accurate system for detecting deepfake speech.

Explanation of each model

1. CNN (Convolutional Neural Network) Model

The CNN model is a baseline deep learning architecture designed to extract spatial and spectral features from input audio spectrograms. It works by applying multiple convolutional layers that detect local patterns, followed by pooling layers that reduce dimensionality while retaining important information.

Convolutional Layers: Capture spectral and frequency-based patterns in the audio spectrogram.

Pooling Layers: Reduce spatial dimensions, making the model computationally efficient.

Fully Connected (Dense) Layers: Perform final classification by mapping extracted features to output labels.

Dropout Regularization: Prevents overfitting by randomly deactivating neurons during training.

Sigmoid Activation: Outputs a probability value (0 or 1) for binary classification (real or deepfake).

2. CNN-MLP (Convolutional Neural Network + Multi-Layer Perceptron) Model

The CNN-MLP hybrid model builds on CNN's feature extraction capability but introduces a deeper Multi-Layer Perceptron (MLP) to enhance classification.

CNN Feature Extractor: The initial layers remain the same as the CNN model, extracting relevant patterns from the spectrogram.

Fully Connected (MLP) Layers: Fully Connected (MLP) Layers: Additional dense layers help the model learn more abstract and complex decision boundaries.

Advantages Over CNN: The added MLP layers increase model expressiveness and improve decision-making by combining multiple learned features.

Use Case: Suitable when raw CNN outputs need further refinement for classification.

3. CNN-LSTM (Convolutional Neural Network + Long Short-Term Memory) Model

The CNN-LSTM hybrid model is designed to capture both spatial features (from CNN) and sequential dependencies (from LSTM).

CNN Layers: Extract key spectral patterns from audio spectrograms. Reshaping Layer: Converts CNN's extracted feature maps into a sequence format suitable for LSTM.

LSTM Layers: Process the sequential data to recognize temporal dependencies, which are crucial in detecting deepfake speech patterns.

Advantages Over CNN: Traditional CNN models do not consider the temporal evolution of features, but LSTM units help capture long-term dependencies, improving detection accuracy.

Use Case: Suitable for detecting subtle changes in speech over time, making it more robust for deepfake audio detection.

3.2 Data Preparation

Dataset source and description: The 'In-the-Wild' Dataset found on Kaggle contains audio recordings from 58 known public figures such as politicians and celebrities. It features a total of 38 hours of audio, partitioned into 20.8 hours of bonafide recordings and 17.2 hours of deepfake recordings. Each speaker contributes an average of 23 minutes of bonafide audio and 18 minutes of spoofed audio. This dataset can be used to evaluate the performance of deepfake detection algorithms as well as complex voice anti-spoofing techniques, especially when considering how well these systems are able to generalize to true in-the-wild recording samples. For further details and access to the dataset, check out the Kaggle page.

Preprocessing techniques:

1. Spectrogram Extraction: First we will extract the Mel-scaled spectrogram of an audio signal. The librosa.feature.melspectrogram function converts the audio signal into a time-frequency representation, emphasizing frequencies relevant to human hearing.

2. Feature Extraction: Here we are going to convert the Mel-spectrogram to a logarithmic scale using librosa.power_to_db, making the dynamic range more manageable. It then computes the Mel-Frequency Cepstral Coefficients (MFCCs) with librosa.feature.mfcc, capturing the timbral texture of the audio. The mean of the MFCCs across time frames is calculated to obtain a fixed-size feature vector for each audio sample.

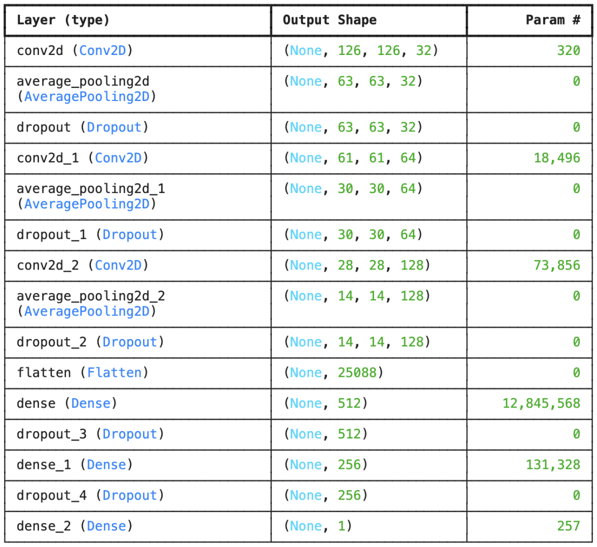
3. Data Preparation: During the data preparation stage, features are extracted from both real and fake audio samples which are stored in separate arrays called realfeatures and fakefeatures. The features are fused into a single dataset called X and the labels Y, indicating whether the audio is real (1) or fake (0). In order to meet the requirements of Convolutional Neural Networks (CNNs) as inputs, each feature vector is resized to a dimension of 128x128 using the method X_resized. Furthermore, another dimension is added to each feature vector to denote a monochromatic color channel, which guarantees that the CNN's input format is respected. These preparations guarantee that the data is properly shaped and structured to efficiently train deep learning architectures.

Train-Test split and validation strategy: The dataset is divided into an 80% training and 20% testing split by setting test_size=0.2. With random_state=38, reproducibility of the split is ensured, and stratify makes sure that the distribution of labels (real vs. fake) is maintained in both the training and testing sets. A validation strategy is also implemented by setting validation_split=0.2, which means that 20% of the training data will be used as validation data for the model during each epoch to measure performance and mitigate overfitting.

3.3 Model Architecture & Training

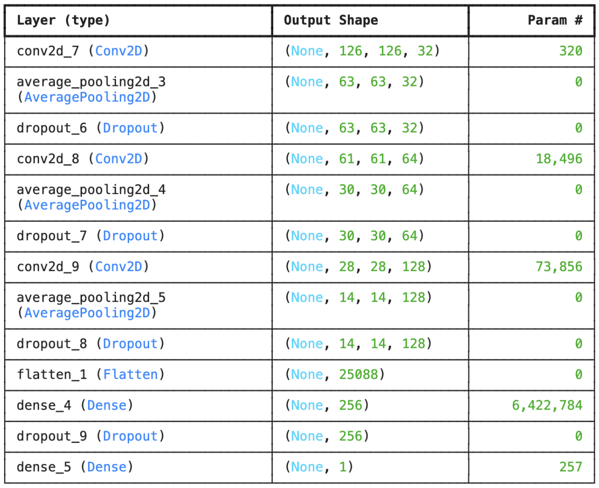
Architectures

CNN-MLP



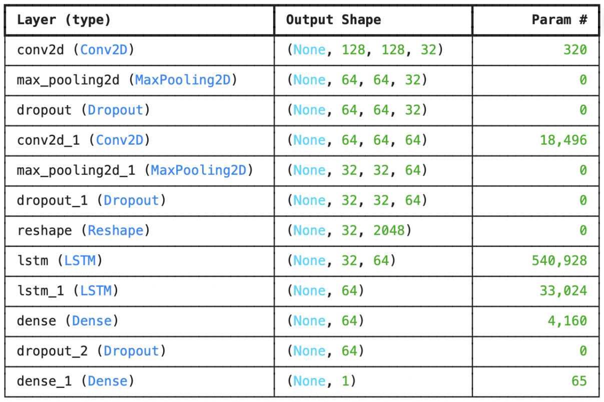
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 126, 126, 32)	320
average_pooling2d (AveragePooling2D)	(None, 63, 63, 32)	0
dropout (Dropout)	(None, 63, 63, 32)	0
conv2d_1 (Conv2D)	(None, 61, 61, 64)	18,496
average_pooling2d_1 (AveragePooling2D)	(None, 30, 30, 64)	0
dropout_1 (Dropout)	(None, 30, 30, 64)	0
conv2d_2 (Conv2D)	(None, 28, 28, 128)	73,856
average_pooling2d_2 (AveragePooling2D)	(None, 14, 14, 128)	0
dropout_2 (Dropout)	(None, 14, 14, 128)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 512)	12,845,568
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 256)	131,328
dropout_4 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 1)	257

CNN



Layer (type)	Output Shape	Param #
conv2d_7 (Conv2D)	(None, 126, 126, 32)	320
average_pooling2d_3 (AveragePooling2D)	(None, 63, 63, 32)	0
dropout_6 (Dropout)	(None, 63, 63, 32)	0
conv2d_8 (Conv2D)	(None, 61, 61, 64)	18,496
average_pooling2d_4 (AveragePooling2D)	(None, 30, 30, 64)	0
dropout_7 (Dropout)	(None, 30, 30, 64)	0
conv2d_9 (Conv2D)	(None, 28, 28, 128)	73,856
average_pooling2d_5 (AveragePooling2D)	(None, 14, 14, 128)	0
dropout_8 (Dropout)	(None, 14, 14, 128)	0
flatten_1 (Flatten)	(None, 25088)	0
dense_4 (Dense)	(None, 256)	6,422,784
dropout_9 (Dropout)	(None, 256)	0
dense_5 (Dense)	(None, 1)	257

CNN-LSTM



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 128, 128, 32)	320
max_pooling2d (MaxPooling2D)	(None, 64, 64, 32)	0
dropout (Dropout)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 64, 64, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 32, 32, 64)	0
dropout_1 (Dropout)	(None, 32, 32, 64)	0
reshape (Reshape)	(None, 32, 2048)	0
lstm (LSTM)	(None, 32, 64)	540,928
lstm_1 (LSTM)	(None, 64)	33,024
dense (Dense)	(None, 64)	4,160
dropout_2 (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65

Training Process The model is compiled with the Adam optimizer set to a binary cross-entropy loss function because the model needs to differentiate between real and fake audio samples, a binary classification problem. The model's performance will be assessed using accuracy. In this case, the model is trained with a batch size of 32, meaning that the weights are refreshed (updated) after every 32 samples are processed (during training). The model trains for a total of 25 epochs, where one epoch denotes using all the training data once. The learning rate of the Adam optimizer is at its default value of (0.001), which works reasonably well for most deep learning tasks. The validation split used is 20%, with 20% of training data held back during each epoch to check how well the model performs and whether it can generalize to new data.

3.4 Challenges Encountered Solutions

1. Handling Audio as Input: This specific project posed a challenge for me because of the audio files analysts with experience handling image and video data struggle with. The barriers of processing and analyzing audio files were new to me. Fortunately, my professor guided me on the principles of audio processing and how to work with strands, STFT (short Time Fourier Transform) representations, and Mel spectrograms using the Librosa audio processing library. This new information on audio processing concepts greatly improved my interpretation of data, allowing me to integrate previously learned skills with working images and advanced algorithms to using microphones and audio files.

2. Feature Selection for Deepfake Detection: One more problem of importance was choosing the appropriate attributes for deepfake audio detection. While talking with my professor, I learned about Mel-Frequency Cepstral Coefficients (MFCC) and its applications, which was a feature of interest in the domain of speech processing. However, extracting MFCC features from the Mel spectrogram posed additional challenges. To overcome this, I spent long hours through various online platforms trying to understand the correct approach to extracting MFCCs from Mel spectrograms with the help of AI assistants. This enabled me to incorporate MFCC features to train my deepfake detection model effectively.

3.5 Assumptions Made

1. Incorporating MLP Layers into CNN: A significant assumption of this study is that the performance of a Convolutional Neural Network (CNN) model would be improved by including Multi-Layer Perceptron (MLP) layers. This assumption is rooted in the belief that MLPs will improve feature enhancement because CNNs are proficient at feature spatial extraction, while MLP layers could enhance processing features for better classification accuracy. The proposed model CNN-MLP aimed to enhance spatial and fully connected feature learning to improve deepfake speech detection.

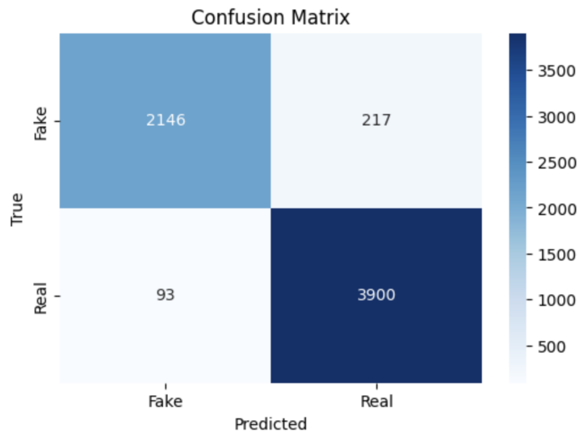
2. Utilizing MFCC Features: Another assumption was that featuring Mel-Frequency Cepstral Coefficients (MFCC) would considerably enhance the model's capacity to identify whether an audio sample was real or fake. Since MFCCs take into account the critical speech features, it was presumed that they would enhance the prediction accuracy for deepfake detection. The experimental results confirmed that this assumption contributed to increased model accuracy.

4 Model Evaluation Analysis

4.1 Performance Metrics

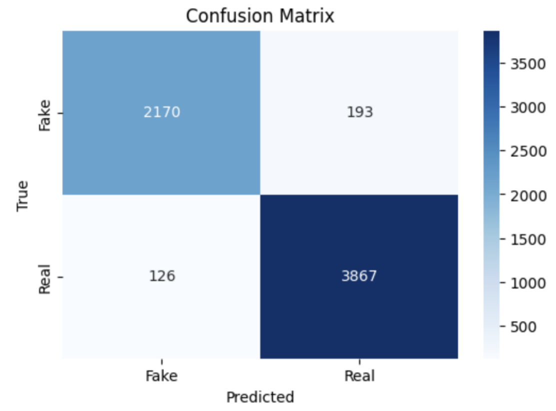
CNN-MPL:

Accuracy: 0.95 Precision: 0.95 Recall: 0.98 F1-Score: 0.96 Loss: 0.19 Confusion Matrix:



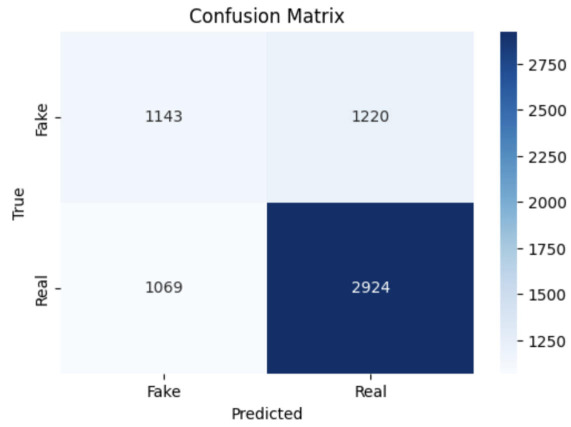
CNN:

Accuracy: 0.95 Precision: 0.95 Recall: 0.97 F1-Score: 0.96 Loss: 0.17 Confusion Matrix:



CNN-LSTM

Accuracy: 0.64 Precision: 0.71 Recall: 0.73 F1-Score: 0.72 Loss: 0.61 Confusion Matrix:



4.2 Observed Strengths Weaknesses

1. CNN-MLP and CNN Achieve Higher Accuracy: The CNN and CNN-MLP models demonstrated superior accuracy and lower loss compared to the CNN-LSTM model. This indicates that these architectures were more effective in learning discriminative features for deepfake speech detection.

CNN Model Has the Fastest Runtime: Among the three models, the CNN model had the fastest runtime. This is primarily because the CNN-LSTM and CNN-MLP models have additional layers, making them computationally more intensive. The CNN model's efficiency makes it a suitable choice for real-time applications.

CNN-MLP Performed Best in Confusion Matrix Analysis: The confusion matrix analysis revealed that the CNN-MLP model had the best performance in terms of distinguishing between real and fake audio samples. This suggests that the addition of MLP layers helped refine feature extraction and classification.

Consistent Improvement in Training Accuracy and Validation Loss: Across all three models, the training accuracy consistently increased over time, indicating effective learning. Additionally, the validation loss showed a steady decrease, demonstrating that the models generalized well to unseen data.

4.3 Comparison with Existing Models

Unlike other models which incorporate data-augmentation techniques deepfake detection models have not yet attempted to use deepfake augmentation, the three models in this study were executed without any form of augmentation. Trying to create models that can be understood and implemented easily was the emphasis. The models are made up of simple blocks which guarantees better performance with less complexity.

These models performed well in deepfake speech detection, achieving reasonable accuracy and low error rates even without data augmentation. Models are easier to use and modify which means most researchers and professionals in the field will be able to adapt them for other uses more easily.

4.4 Suggestions for Future Improvements

The model's robustness and generalizability in real-world applications can be improved by training using diverse datasets from multiple sources. In addition, the model can be further refined by incorporating advanced feature extraction techniques beyond MFCC like chroma features or spectral contrast.

As the described model serves as a foundational framework for advanced deepfake detection, augmenting the model's architecture could optimize the model's performance while also improving the baseline. Focusing on incorporating data augmentation techniques in the models will aid in enhancing their generalization ability.

5 Reflection Discussion

5.1 Key Challenges in Implementation

One of the challenges that I met during the implementation process was the congestion due to the excessive time needed for model training. On average, the models required more than four hours for completion even when restricting the number of epochs to 25. This limit constrained training interval or exploding epoch counts or experimenting with multiple datasets.

For future improvement, the problem can be resolved by the availability of high-performance GPUs. With adequate resources, more extensive training sessions can be undertaken, aiding the models in learning from various datasets. This measure would improve model accuracy, change tolerance and overall system efficiency, heightening deepfake detection capability.

5.2 Real-World Applicability

The constructed models extend beyond the bounds of controlled research datasets and show efficacy in real-world applications. In assessing practicality, the models were able to correctly classify both an AI-generated deepfake voice and a real human voice sample. This suggests the models are able to surpass the confines of research settings and contend with actual variations in audio data.

This adaptability makes the models appropriate for use in media authentication, cyber defense, and forensic investigations. The models' ability to detect speech deep fakes in unconstrained settings indicates that they could be used in real-time monitoring systems, improving reliability and confidence regarding voice authentication and communication systems.

5.3 Potential Improvements with Additional Data or Resources

In order to improve the model's performance, it is necessary to obtain more diverse and up-to-date datasets. Adding datasets like ASVspoof could help improve the model's accuracy in deepfake audio detection. Moreover, including multi-lingual speech samples MADD offers would enhance the model's deepfake speech detection capabilities, thereby strengthening its relevance in many global regions.

In addition, the model's predictive accuracy would be greatly improved with training on larger data sets. However, processing large datasets quickly and efficiently requires advanced computational resources like high-end

GPUs for improved training time and optimized computation. With these additional resources, the model could be accurately configured for practical application in deepfake detection across various global languages and acoustics.

5.4 Deployment Considerations

We start by training the model, after which we proceed to save it. With the model saved, it can now be utilized in other systems that require the use of deepfake detection like a voice call deepfake detection application. The application will use the saved model as the main component for prediction purposes. Through model saving, deepfake detection can be implemented in different environments, documents, and applications, which increases its reuse potential and flexibility. Such an approach ensures that audio can be analyzed in real-time and provides comprehensive solutions to deepfakes detection across multiple systems.

6 Conclusion

This work is geared toward the construction of deepfake detection models for audio, specifically for the detection of deepfake voice calls. The objective was to apply various architectures such as CNN, CNN-MLP, and CNN-LSTM to determine which structure suffers less in terms of detecting deepfake audio. The best results in accuracy and loss were yielded by the CNN and CNN-MLP models, while the worst performing model was the CNN-LSTM. The results of the CNN-MLP model were especially good.

The models employed Mel spectrograms along with the MFCC features derived from the audio data to capture relevant patterns and characteristics. The assumption that better performance results will stem from adding MLP layers to the CNN structure was indeed correct, thus improving overall accuracy. Moreover, the results from adding the MFCC features also positively impacted model performance.

Despite not using data augmentation, our models were robust relative to other advanced approaches in deepfake detection. The models' clear-cut logic and ease of use render them a rich source of accessible information to build on and further develop toward deployment in real-life scenarios.

Subsequent work could use such techniques as data augmentation for improving model performance, training on datasets with different demographics, multilingual datasets, and other methods to improve model generalization. Implementing these models in an installation for the purpose of real-time deployment is suggested.

References

[1] Pham, Lam, et al. ”A Comprehensive Survey with Critical Analysis for Deepfake Speech Detection.” arXiv preprint arXiv:2409.15180 (2024).

[2] Yu, Xiaomin, et al. ”Fake artificial intelligence generated contents (faigc): A survey of theories, detection methods, and opportunities.” arXiv preprint arXiv:2405.00711 (2024).

[3] Li, Menglu, Yasaman Ahmadiadli, and Xiao-Ping Zhang. ”Audio anti-spoofing detection: A survey.” arXiv preprint arXiv:2404.13914 (2024).

[4] Croitoru, Florinel-Alin, et al. ”Deepfake Media Generation and Detection in the Generative AI Era: A Survey and Outlook.” arXiv preprint arXiv:2411.19537 (2024).

[5] Li, Yupei, et al. ”From Audio Deepfake Detection to AI-Generated Music Detection–A Pathway and Overview.” arXiv preprint arXiv:2412.00571 (2024).

[6] Zou, Yueying, et al. ”Survey on AI-Generated Media Detection: From Non-MLLM to MLLM.” arXiv preprint arXiv:2502.05240 (2025).