Audio Deepfake Detection

1 Introduction

An audio deepfake is a type of synthetic audio that uses artificial intelligence (AI) to mimic a person's voice so convincingly that it sounds like that person actually said something they didn't. It's a subset of deepfake technology, which also includes manipulated images and videos.

Audio deepfakes have both beneficial and harmful use cases. On the positive side, they are used in entertainment for dubbing movies, reviving voices of deceased actors, and creating realistic voiceovers in games and audiobooks. They also help people with speech impairments regain a personalized voice. On the negative side, audio deepfakes can be misused for fraud, such as impersonating executives in scams, spreading misinformation, or creating fake recordings to damage reputations. While the technology offers creative and accessibility benefits, it also poses serious ethical and security risks.

2 Methods and Models

There are many methods and models to develop audio deepfakes, prominent among them are:

2.1 Convolution Neural Networks

CNNs can be used to detect audio deepfakes, CNNs is used by analyzing the spectrum features of audio. CNNs uses local spatial features for classification of real and fake audio. By using an efficient neural network, performance accuracy of 99% can be achieved which is better than any other models.

CNNs have very high accuracy, also CNNs are very fast at inference because of which it can be used in real-time usage which makes it suitable for end-to-end pipeline on real conversation.

While there are many advantages, CNN requires well preprocessed spectrograms and it might struggle when unseen spoofing technique is used.

2.2 Bidirectional LSTM

Bidirectional LSTM captures sequential dependencies in audio data to detect the real and fake audio by modeling natural speech patterns and inconsistencies.

BiLSTM achieves accuracy of 98% which is slightly lesser than CNNs.

Bidirectional LSTMs can model the context in both directions which makes it powerful for audio which are time dependent, because of which it can be tuned for real-time processing with smaller windows.

BiLSTMs have higher latency than CNNs because it processes the data sequentially and due to which training time becomes long.

2.3 Quantum SVM

Q-SVM uses Quantum Kernel Support Vector Machines, uses quantum kernel estimation to capture complex features spaces of the dataset.

When tuned perfectly, can achieve performance of 99%.

Q-SVMs are robust to feature noise which makes it perfect for noisy or degraded data, also SVMs are lightweight compared to others once trained.

But to use Q-SVM we need feature engineering, and its implementation may require special libraries which are tough to understand.

2.4 Selected Approach

I selected the CNN-based approach due to:

- High accuracy on benchmark datasets
- Real-time potential with fast inference
- Compatibility with mel-spectrogram as inputs which is considered best suited for audio clips processing

3 Implementation

- First divided the audio into two separate folders one with tampered and other with untampered audio all in the .wav format
- The dataset was preprocessed using librosa library to extract mel-spectrogram features which will be given as input to the model
- Designed an effective 2D CNN architecture which will classify tampered and untampered data.
- Divided the data into training and testing data in the ratio of 80:20 which was used to train and evaluate the model

3.1 Challenges

1. The length of the audio were different which were making the data inconsistent to overcome this I cropped the audio to 3s or we could use padding also.
2. The size of the spectrogram mismatched, standardized the time-axis using zero-padding.

3.2 Assumptions

- All .wav files are valid human speech samples
- 3 seconds of audio is sufficient to capture manipulation artifacts

4 Analysis

Convolution Neural Networks are proved to be highly effective when used in image-like data and mel-spectrogram which turns audio into 2D visual patterns. One of the major advantages of using CNNs is its balance between speed and performance.

Its architecture is lightweight as it doesn't require heavy preprocessing pipelines and also supports real-time detection. It can easily be integrated into edge devices on the go.

The CNN model takes the Mel-spectrogram of the images i.e. the visual representation of the audio as its inputs. The model captures time 128 (frames) and frequency 64 (Mel bands).

- The first layer of the model uses filters to extract the low level features like pitch patterns and harmonics, ReLU activation was used which introduced non-linearity to learn complex patterns.
- The second layer learns high level features like temporal voice again followed by ReLU.
- Maxpooling and dropout layers were added to reduce the spatial dimension of the feature maps which helps in focusing on the most important features of the image and dropout is used to prevent overfitting by dropping some neurons in each iteration.
- Flatten layers were used to convert the 2D output of the layer into 1D vector to be used as input to Dense layer, which combines and interprets the features of the images which are learned by CNN layers.
- The final Dense layer uses sigmoid activation, which outputs values between 0 and 1, which can be used to classify image as real or fake.

4.1 Strengths

1. Simple
2. Uses mel-spectrogram, which are robust
3. Balance between speed and accuracy

4.2 Weaknesses

- May miss subtle manipulations in higher-quality fakes
- Doesn't model temporal context well (unlike RNNs or transformers)
- Performance may drop in highly noisy or multilingual environments

4.3 Future Improvements

1. Attention mechanism or temporal model can be added
2. Multilabel classification can be done to differentiate among different tampering types
3. Pretrained models can be used like wav2vec 2.0

5 Reflection Questions

5.1 What were the most significant challenges in implementing this model?

The most significant challenge is interpreting the result without having any benchmark.

5.2 How might this approach perform in real-world conditions vs. research datasets?

The model performs very well on research datasets but in real world conditions it may not be perfect as there are many background noises. It may not be able to include accent, emotions into decisions making. It would require fine tuning with the real-world wild data.

5.3 How would you approach deploying this model in a production environment?

By exporting the model as TFLite to be used in mobile applications and similar devices, or integrate it into a streaming audio pipeline.