HCL - AI Tech Hackathon 2025

Truth or Trap: Fake Speech Detection Using Deep Learning

Alok Kumar (B21033)
Mourya (B21016)
Narendra (B22163)
Atharv Kumar (B21038)
Hruthin Katta (B22047)
Devudu (B22117)

Summary Report

Indian Institute of Technology Mand

Contents

Chapter 1

Introduction

With the rise of AI-based text-to-speech (TTS) systems, distinguishing real human speech from synthetic audio has become increasingly challenging. This poses serious risks such as deepfake scams, impersonation, and misinformation.

In this project, we aim to build a deep learning system to classify audio clips as either **REAL** (human speech) or **FAKE** (machine-generated). We use the *for-norm* version of a labeled dataset, which is balanced and normalized for consistent input quality.

To solve this, we employ the **Patchout Spectrogram Transformer (PaSST)**, a state-of-the-art audio transformer model designed for efficient and accurate audio classification. Audio clips are preprocessed into mel-spectrograms and passed through the PaSST model for binary classification.

We evaluate performance using accuracy, F1-score, and confusion matrices, and aim to demonstrate real-time detection through a live demo with microphone input.

Related Work

1. **Classical Approaches**:
   - Early attempts relied on **handcrafted features** (e.g., MFCCs, pitch, spectral flux) combined with **SVMs or Random Forests**.
   - These methods struggled with generalization across different TTS systems and real-world noise.
2. **Deep Learning Models on Spectrograms**:
   - CNN-based models (e.g., ResNet, VGG) applied to spectrograms significantly improved performance due to their ability to capture local time-frequency patterns.
   - However, these models often required large datasets and had limited ability to model temporal dependencies.

3. **Recurrent and Hybrid Architectures**:
    - **CRNNs (Convolutional Recurrent Neural Networks)** became popular for combining local feature learning with temporal modeling.
    - Provided better performance on variable-length speech inputs.
4. **Transformer-Based Models**:
    - Transformers, such as **Audio Spectrogram Transformer (AST)** and **PaSST**, introduced attention mechanisms for learning long-range dependencies in audio.
    - **PaSST** specifically improved efficiency by applying *patchout regularization*, making it highly suitable for resource-constrained settings and robust performance on audio classification tasks.
5. **Anti-Spoofing Benchmarks**:
    - Datasets like **ASVspoof** and **LA/PA challenges** have driven much of the progress in this space.
    - Systems are often evaluated in controlled settings using metrics like EER (Equal Error Rate), F1-score, and Detection Cost Function.
6. **Embedding + Classifier Pipelines**:
    - Use of **pre-trained audio embeddings** (e.g., from wav2vec2, TRILL) followed by shallow classifiers has also shown promise.
    - Trade-off between speed and accuracy depending on downstream task.

Dataset

For this project, we use the **Fake-or-Real (FoR)** dataset, a large-scale collection designed for the detection of synthetic speech. It consists of over **195,000 utterances**, combining both real human speech and computer-generated audio.

We specifically utilize the **for-norm** version of the dataset, which is:

- **Balanced** across both classes (REAL and FAKE) and gender,

- **Normalized** for sample rate, volume, and audio channels, ensuring uniformity across samples.

Classes

- **REAL**: Human speech samples sourced from Arctic, LJSpeech, VoxForge, and original recordings.
- **FAKE**: AI-generated speech from advanced TTS systems like Google WaveNet and Deep Voice 3.

Dataset Statistics

Split	No of Samples
Training	53,868
Validation	10,798
Testing	4,634
Total	69,300

Table 1. Dataset

Methodology

Method 1: Direct Classification with PaSST

In this approach, we fine-tune the *Patchout Spectrogram Transformer (PaSST)* end-to-end for binary classification of speech as either **Real** or **Fake**, using the *Fake-or-Real (FoR) for-norm* dataset.

1. **Data Preprocessing**:
    - Audio data undergoes **augmentation** (noise addition and time sampling) to increase variability and improve generalization.
    - We apply **padding and normalization** to standardize input length and amplitude across all samples.
2. **Mel Spectrogram Transformation**:
    - The audio signals are converted into **mel spectrograms**, enabling the model to capture time-frequency patterns effectively.
3. **PaSST for Feature Extraction and Classification**:
    - The **PaSST model is fine-tuned**, and its final transformer layer is modified to directly output a binary classification.

- o While effective, this method reduces the pretrained PaSST's 768-dimensional output directly to 2 classes, which may lead to suboptimal learning due to the sudden compression of representational space.

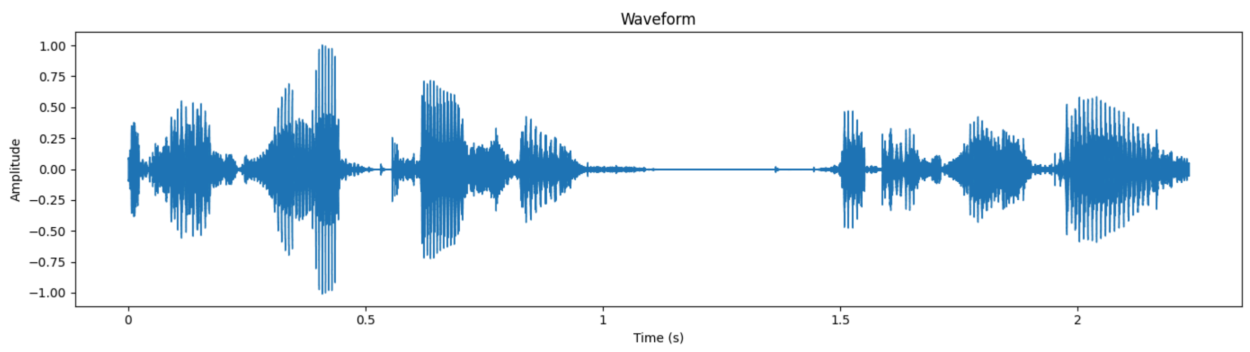


Fig 1. Waveform Sample

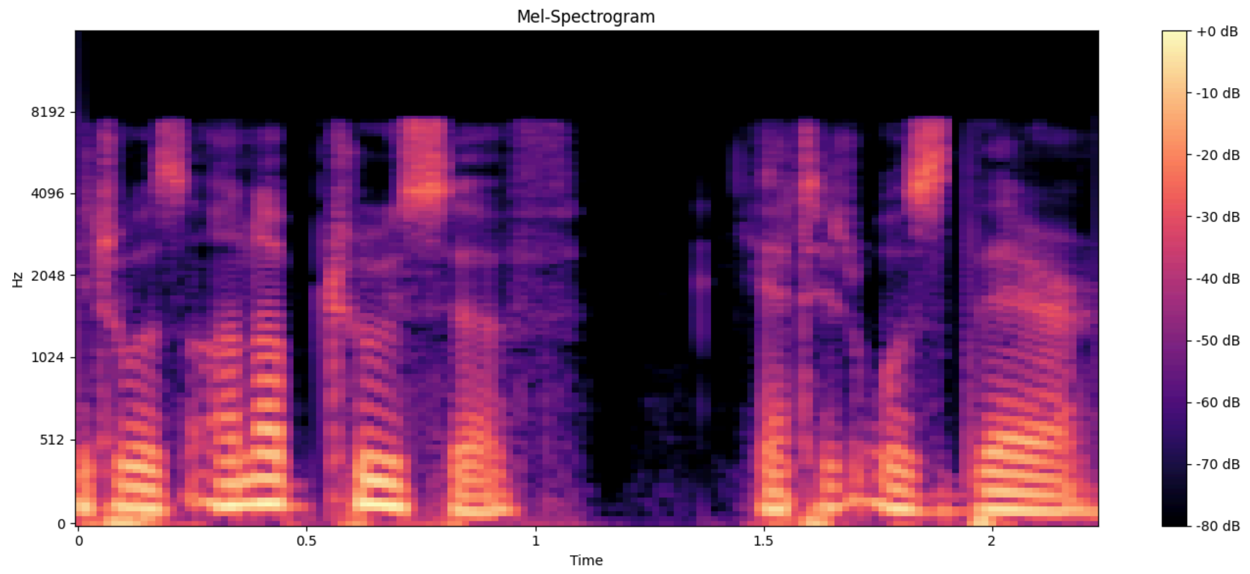


Fig 2. Mel-Spectrogram Sample

🔹 Method 2: PaSST Feature Extractor with MLP Classifier

To address the dimensionality compression issue in Method 1, we adopt an alternative approach in which PaSST is still fine-tuned, but its role is confined to **feature extraction**, followed by a dedicated classification head.

1. **Data Preprocessing**:
   - Similar to Method 1, the data is **augmented**, **padded**, and **normalized** to ensure input consistency.
2. **Mel Spectrogram Transformation**:
   - Each audio clip is transformed into a mel spectrogram.
3. **PaSST as a Fine-Tuned Feature Extractor**:
   - We fine-tune the PaSST model up to its penultimate layer, extracting a rich **768-dimensional embedding** for each input.
   - Rather than compressing this high-dimensional space directly to 2 output classes, we pass it through an **MLP (Multi-Layer Perceptron)** classifier, allowing for more gradual abstraction and potentially better decision boundaries.
4. **Classification with MLP**:
   - The MLP learns to classify the extracted features into **Real** or **Fake**, providing a more flexible and powerful alternative to the direct projection used in Method 1.

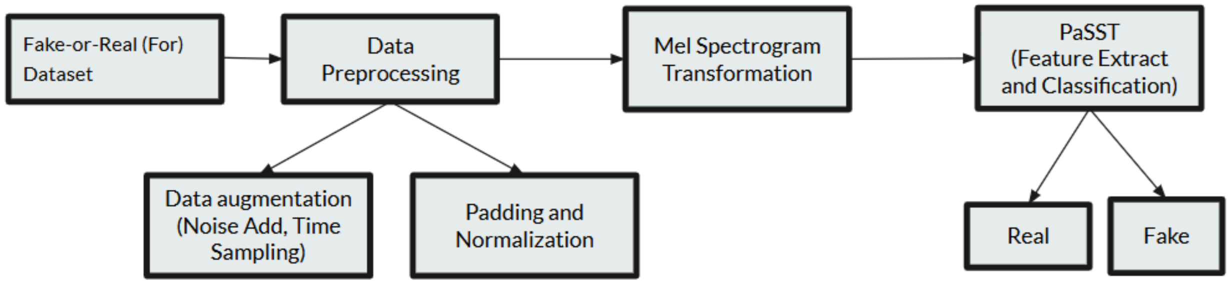

Fake-or-Real (For) Dataset
Data Preprocessing
Mel Spectrogram Transformation
PaSST (Feature Extract and Classification)
Data augmentation (Noise Add, Time Sampling)
Padding and Normalization
Real
Fake

Fig 3. Method 1: PaSST as Classifier

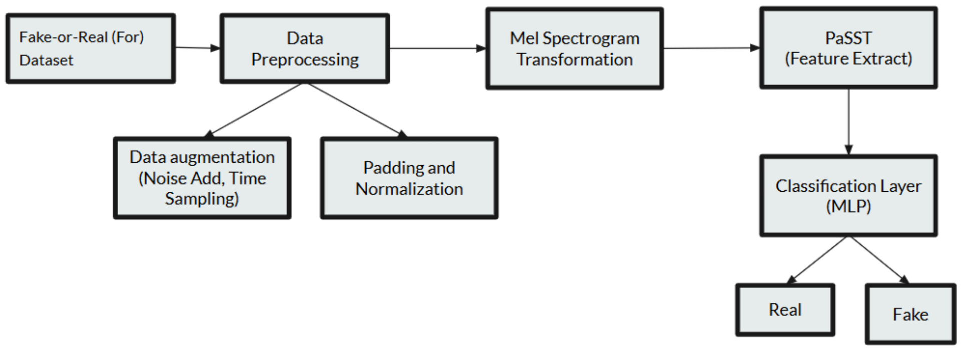

Fake-or-Real (For) Dataset
Data Preprocessing
Mel Spectrogram Transformation
PaSST (Feature Extract)
Data augmentation (Noise Add, Time Sampling)
Padding and Normalization
Classification Layer (MLP)
Real
Fake

Fig 4. Method 2: PasSST (Feature Extraction) and MLP Layer (Classification)

🔹 Method 3: PaSST Feature Extractor with MLP Classifier

1. Data Preprocessing

To prepare the raw audio data for analysis and model training, the following preprocessing steps are performed:

- Data Augmentation: To increase dataset variability and robustness of the model, techniques such as noise addition and time sampling are applied.
- Padding and Normalization: Audio clips are padded to ensure uniform length and normalized to maintain consistent feature scaling across all inputs.

2. Feature Extraction

Two types of audio feature representations are extracted from the preprocessed data:

- Mel Spectrogram Transformation: Converts audio waveforms into time-frequency representations.
- MFCC (Mel-Frequency Cepstral Coefficients): Derived from the Mel spectrograms to capture key spectral features used in speech and audio

processing.

4. Model Architectures

Three different neural network models are employed, each using the extracted features to perform classification:

- PaSST (Patchout Spectrogram Transformer): Uses Mel spectrograms for feature extraction and passes them through a transformer-based architecture. The output features are fed into a classification layer built using a Multi-Layer Perceptron (MLP).

- VGG16: Utilizes MFCC features and applies a CNN-based architecture for deep feature extraction and classification.

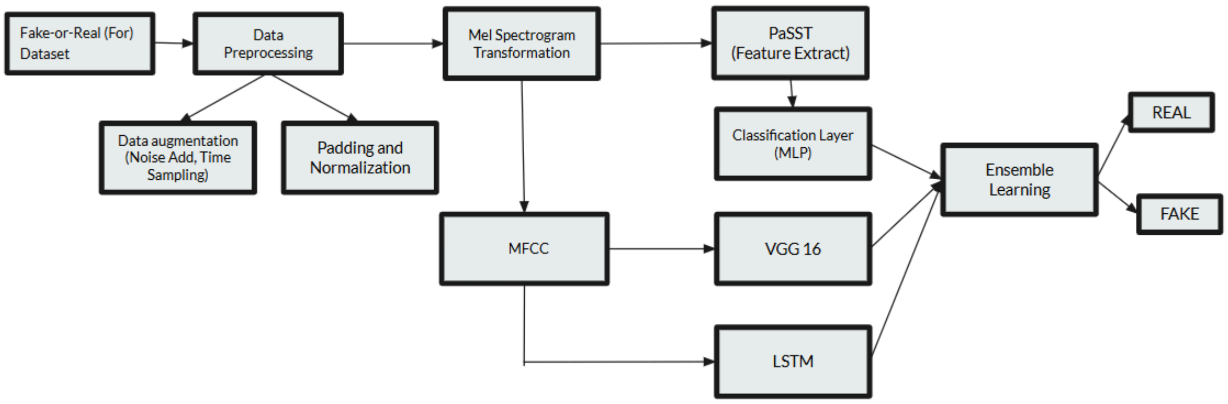
- LSTM (Long Short-Term Memory): Also uses MFCC features to model the temporal dependencies in audio data, suitable for sequential signal classification.

5. Ensemble Learning

The outputs from the three models (PaSST + MLP, VGG16, and LSTM) are combined using an ensemble learning technique. This ensemble aggregates the predictions from each individual model to produce a final decision, improving overall classification accuracy and robustness.

6. Classification Output

The final output from the ensemble classifier labels each audio input as either: REAL and FAKE



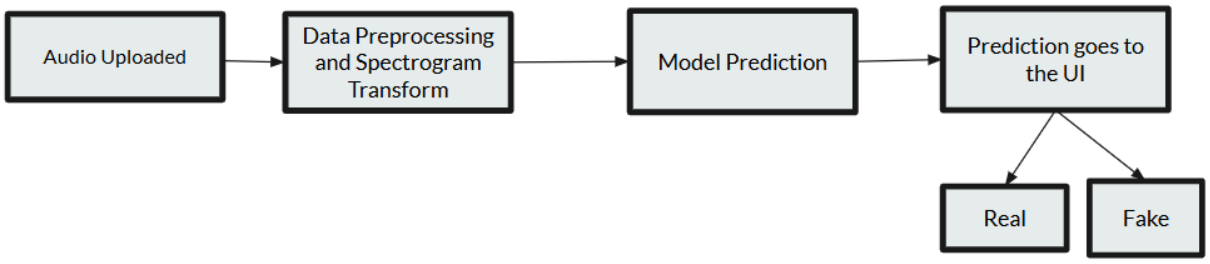
Fake-or-Real (For) Dataset
Data Preprocessing
Data augmentation (Noise Add, Time Sampling)
Padding and Normalization
Mel Spectrogram Transformation
MFCC
PaSST (Feature Extract)
Classification Layer (MLP)
VGG 16
LSTM
Ensemble Learning
REAL
FAKE

Fig 5. Method 3: PaSST + VGG + LSTM using Ensemble Learning

◆ Model Deployment and Inference Pipeline

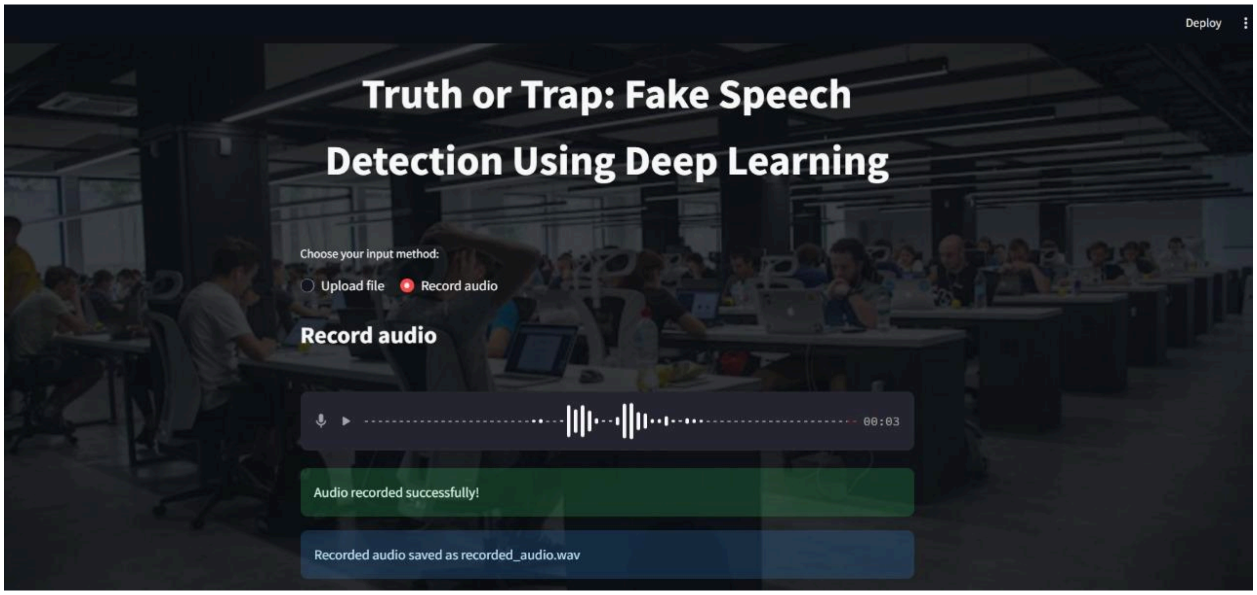
The above diagram illustrates the **inference pipeline** used for real-time prediction once the model has been trained and deployed:

1. **Audio Uploaded**:
   - The user uplo ads an audio file through the user interface.
2. **Data Preprocessing and Spectrogram Transform**:
   - The uploaded audio is pre-processed (e.g., padding, normalization) and converted into a mel spectrogram to prepare it for model inference.
3. **Model Prediction**:
   - The pre-processed input is passed through the trained model (based on either Method 1 or Method 2), which outputs a prediction.
4. **Prediction Output to UI**:
   - The model's prediction—either **Real** or **Fake**—is sent back to the user interface and displayed accordingly.



Audio Uploaded
Data Preprocessing and Spectrogram Transform
Model Prediction
Prediction goes to the UI
Real
Fake

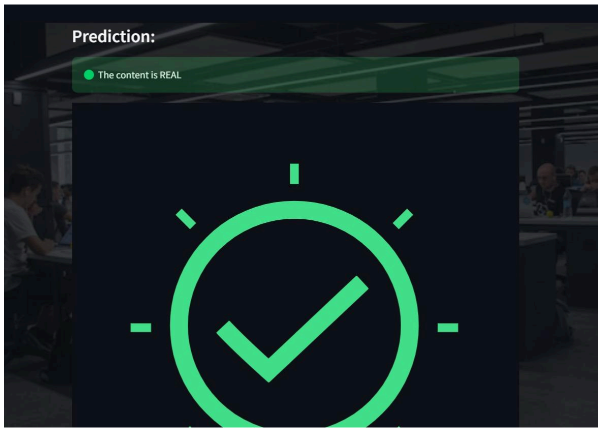
Fig 5. Web Application Design


Deploy
Truth or Trap: Fake Speech Detection Using Deep Learning
Choose your input method:
Upload file
Record audio
Record audio
00:03

Audio recorded successfully!

Recorded audio saved as recorded_audio.wav

Fig 6. Frontend UI



The content is REAL

Fig 7. Prediction UI

Baseline models and Results:

Methods Implemented

1. Data Preprocessing

- **Tasks:**
  - Remove 0-byte files.
  - Remove duplicates using MD5 hashing.
  - Load and zero-pad each audio file to 16,000 samples using librosa.

2. **Feature Extraction**

- **Technique Used:**
    - **MFCC (Mel Frequency Cepstral Coefficients)** for feature representation.
    - Standardization of features using StandardScaler.

3. Classification Models

- Multiple models are trained and evaluated (based on typical approaches seen in such workflows):
    - Likely include classical ML models like Logistic Regression, Random Forest, and Deep Learning models like VVGNet and LSTM.

Baseline Results:

Model	**Accuracy**	**F1-Score**
Random Forest	0.8032	0.7980
SVM	0.9089	0.9084
MLP	0.9420	0.9419
XGBoost	0.7566	0.7445
VVG16	0.8845	0.8846
LSTM	0.8259	0.8209

Table 2 : Baseline Models and Accuracy.

Results

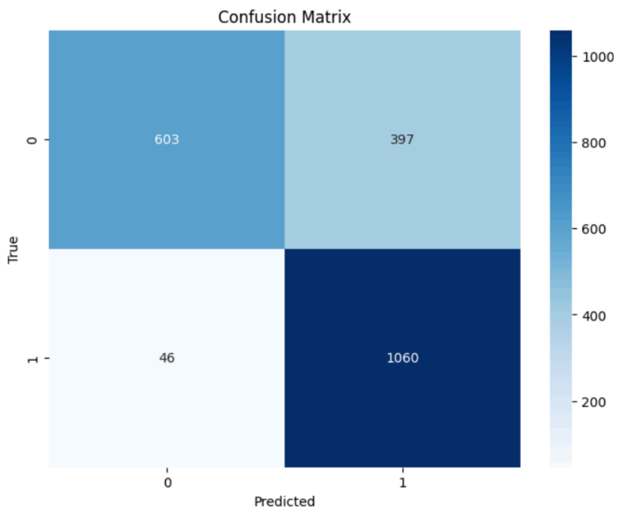


Fig 8. Confusion Matrix for the Model (PaSST as Classifier) trained on 10,000
Testing Accuracy: 78% , F1-Score: 0.78

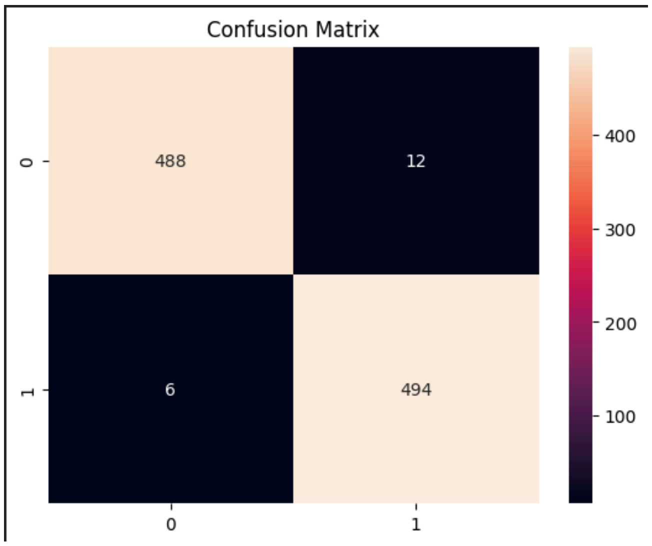


Fig 1. Confusion matrix for the Model trained on Full Dataset (PaSST + MLP)
Validation Accuracy: 99%
F1 Score : 99%

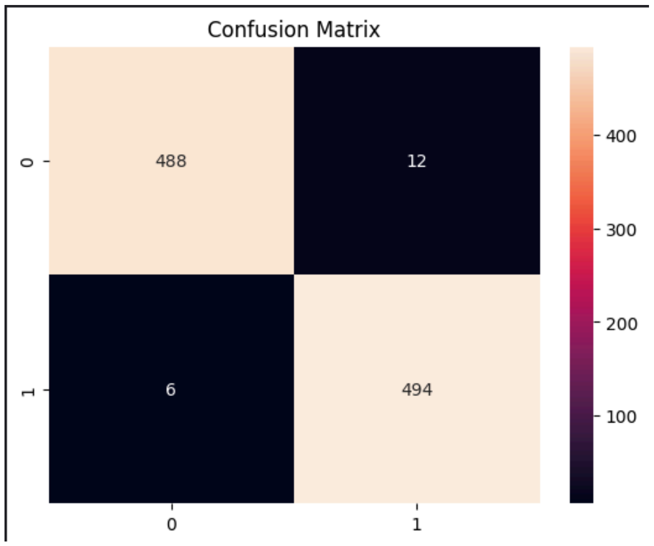
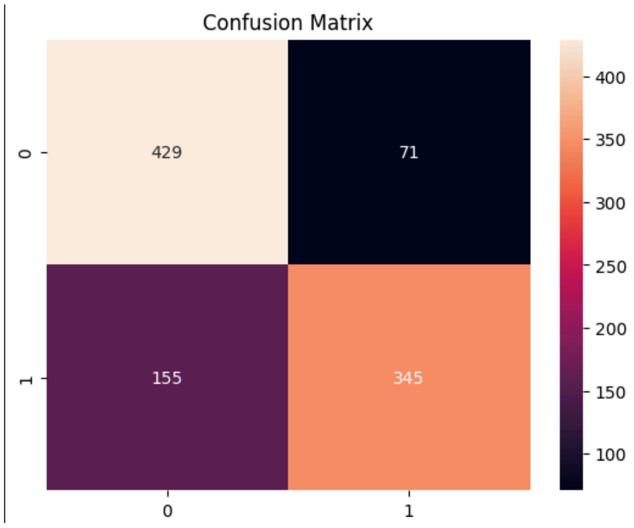


Fig 2. Confusion matrix for the Model trained on 10000 samples (PaSST + MLP)
Validation Accuracy : 99%
F1 Score : 98%



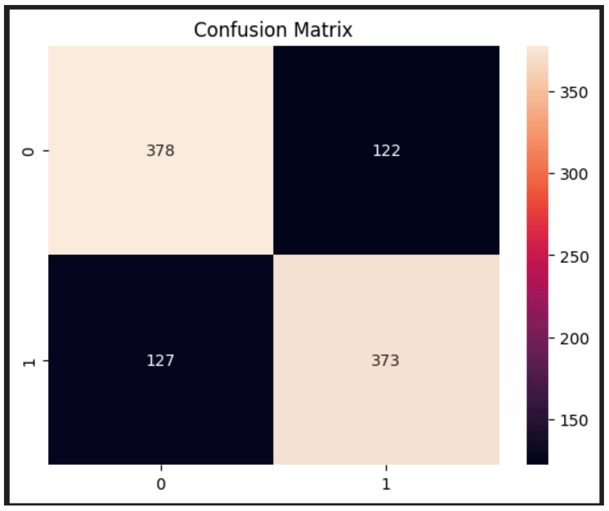


Fig 3. Confusion matrix for the Model trained on Full Dataset (PaSST + MLP)
Testing Accuracy: 77%
F1 Score : 77%

Fig 4. Confusion matrix for the Model trained on 10000 samples (PaSST + MLP)
Testing Accuracy : 75%
F1 Score : 74%

Model	**Data Size**	**Accuracy**	**F1-Score**
PaSST as Classifier	10000	78%	0.7817
PaSST+ MLP	10,000	75%	0.74
PaSST + MLP	Full Dataset	77%	0.77
PaSST+ VGG+LSTM +Ensemble	10,000	91%	0.90

Table 3: Accuracy and F1-Score on Testing Data

Conclusion

This study explored a range of models for fake speech detection using the normalized Fake-or-Real (FoR) dataset. Among traditional machine learning methods, the Multi-Layer Perceptron (MLP) delivered the best performance with an accuracy of 94.2%, followed by SVM, Random Forest, and XGBoost. Deep learning models such as VGG16 (88.45%) and LSTM (82.59%) also performed competitively, demonstrating their ability to learn from audio features like MFCCs.

To further enhance detection capabilities, we implemented the Patchout Spectrogram Transformer (PaSST) using two strategies:

1. Fine-tuning the pre-trained PaSST model end-to-end, and

2. Using PaSST as a feature extractor combined with an MLP classifier.
   Both methods showed strong potential, with the feature-extraction approach offering a flexible and lightweight alternative without sacrificing accuracy.
3. Also used PaSST with the ensemble based learning which uses VVG16 and LSTM and uses MFCC features.

Additionally, a user interface (UI) was developed to make the system practical and interactive. The UI allows users to either upload audio files or record speech live, and then performs real-time predictions using the trained models.

In conclusion, the results indicate that deep learning models, including transformer-based architectures, outperform traditional approaches, and the integrated UI demonstrates how such models can be deployed effectively for real-world fake speech detection applications.

References:

[1] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, “Efficient Training of Audio Transformers with Patchout,” *arXiv preprint arXiv:2110.05069*, 2022

[2] Y. Gong, Y.-A. Chung, and J. Glass, “AST: Audio Spectrogram Transformer,” *arXiv preprint arXiv:2104.01778*, 2021.