CS 671

2nd to 4th May

Truth or Trap: Fake Speech Detection Using Deep Learning

By: *GROUP 43:*

Atharv Kumar (B21038)
Alok Kumar (B21149)
Mourya Kondawar (B21016)
Hruthin Katta (B22047)
Narendra (B22163)
Devudu (B22117)

Table of Contents

Introduction

- The quick development of Text-to-Speech (TTS) combined with AI voice cloning technologies has made synthetic voices indistinguishable from natural human voice patterns.
- The great advantages delivered by assistive technology and virtual assistants and content creation tools come with dangers from audio deepfakes and impersonation schemes and incorrect information dissemination.
- Evaluating synthetic speech stands as the basis for maintaining digital trust.

Objectives

- Develop a deep learning model to accurately classify speech as REAL (human) or FAKE (AI-generated) using a balanced, normalized dataset.
- Implement an end-to-end pipeline with audio preprocessing, mel-spectrogram conversion, training, evaluation (F1-score, confusion matrix), and embedding visualization (t-SNE/PCA).
- Deploy a real-time demo using Streamlit or Gradio that enables live microphone-based fake speech detection.
- Develop a robust environment for audio fake detection which can record voices in real time and classify if it is fake or not.

Problem Statement

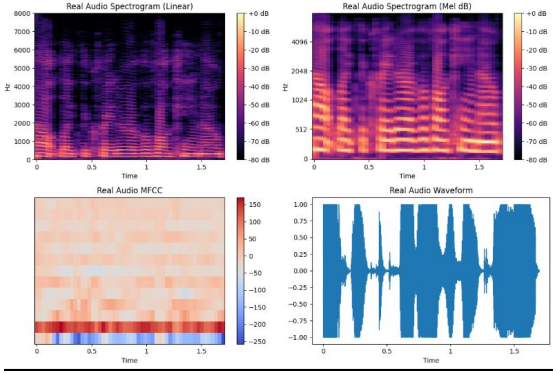
- The system should deploy deep learning algorithms to distinguish sound files into REAL recordings from genuine human communication and FAKE audio that TTS artificial intelligence techniques produce. The system must demonstrate high performance on real-time audio text with low weight and reliable operation.
- The integration involves receiving real speech samples along with recordings synthesized by AI Text-to-Speech technologies.
- The training data used to build the classification model should be the normalized “for-norm” dataset which has been purposefully balanced by equal gender representation and class composition and properly normalized in terms of audio rate frequency and volume together with channel representation.

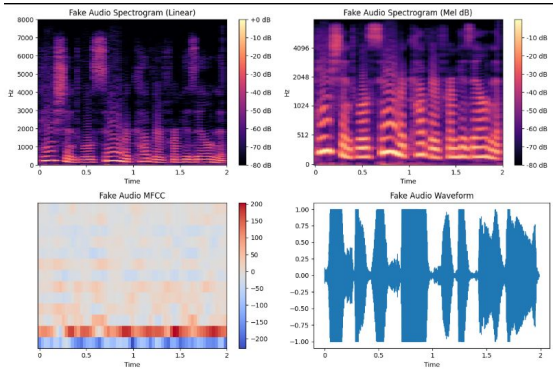
Dataset

- We are using FoR(Fake Or Real) Dataset:
  - We have used the Version: **for-norm audio dataset**
  - It consists of over **195,000 utterances**, combining both real human speech and computer-generated audio.

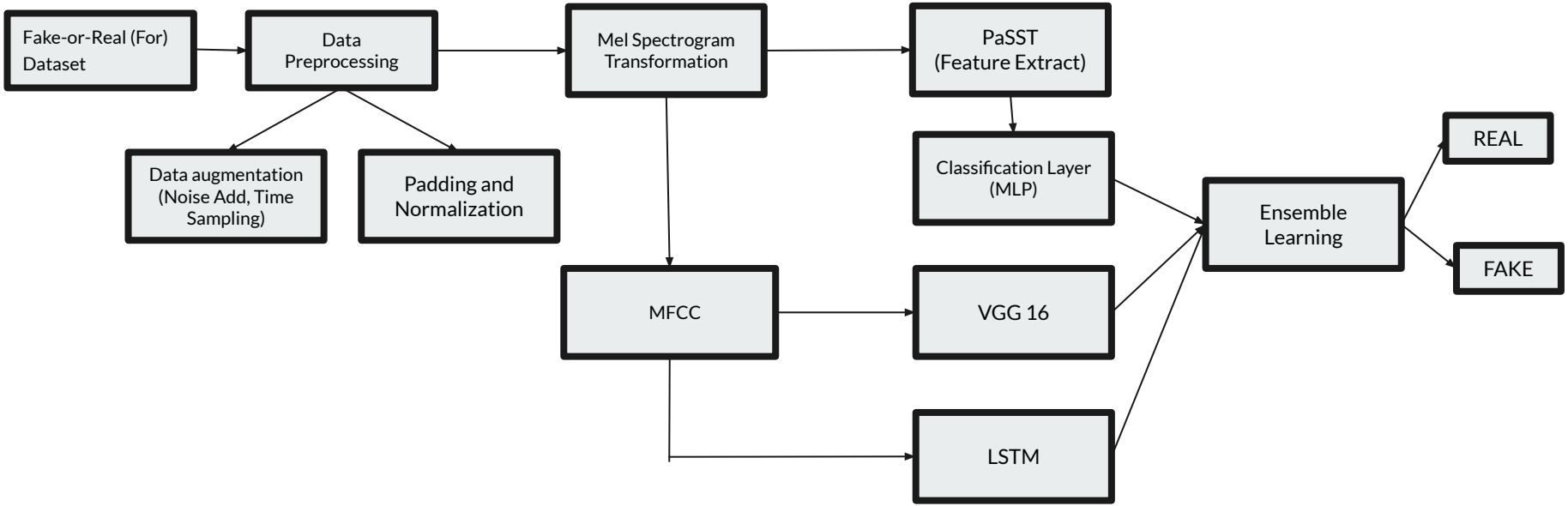
Characteristics:

- Balanced by gender and label class.
- Normalized for sample rate, volume, and number of channels.
- Splits: Train, Validation, Test.
- Labels: **REAL**, **FAKE**.

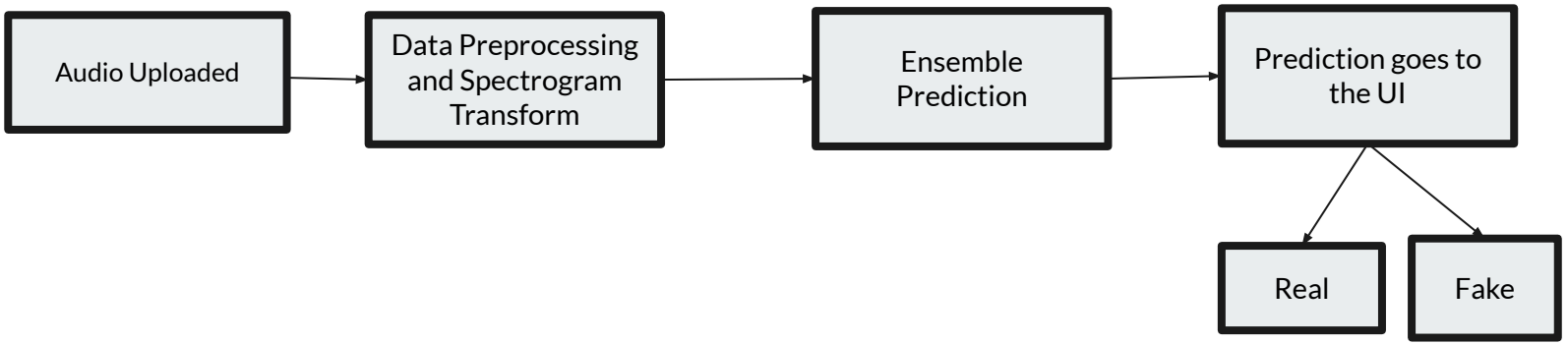




Overall Model

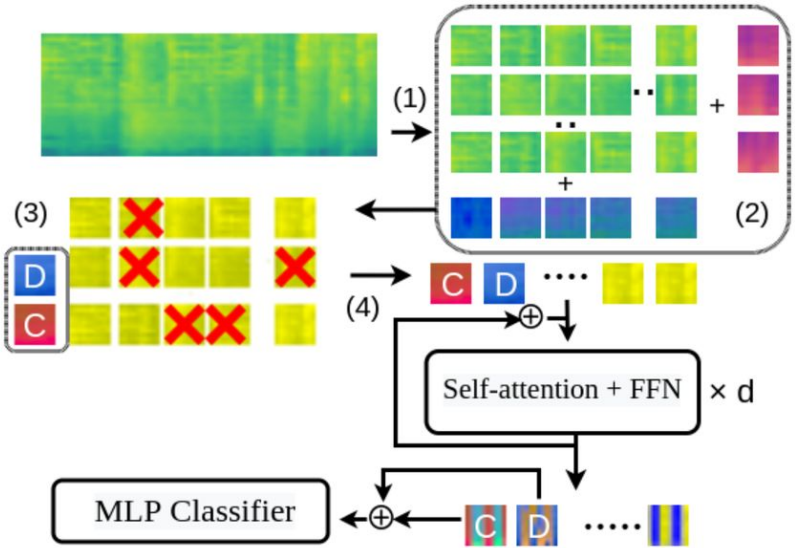


Web Application Design



METHODOLOGY

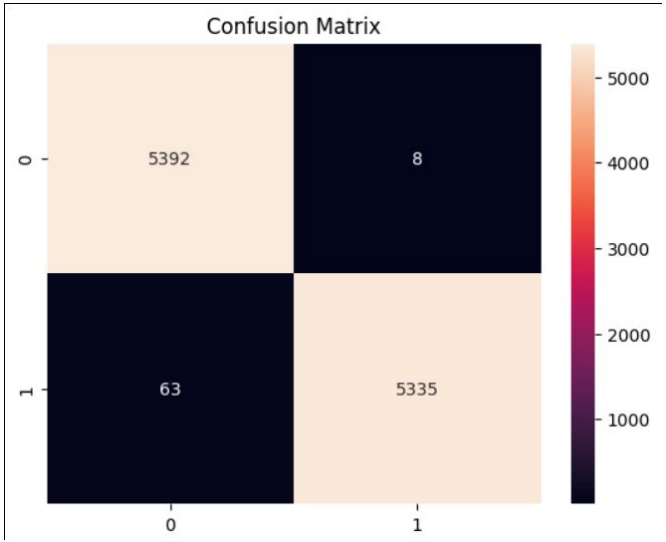
- **PaSST (Patchout Audio Spectrogram Transformer)**, a state-of-the-art transformer-based model designed specifically for audio classification tasks. It operates on mel-spectrograms and leverages patch-based attention with patchout regularization to enhance generalization while reducing computation. We have added MLP layers for classification. (regular downscaling of features)



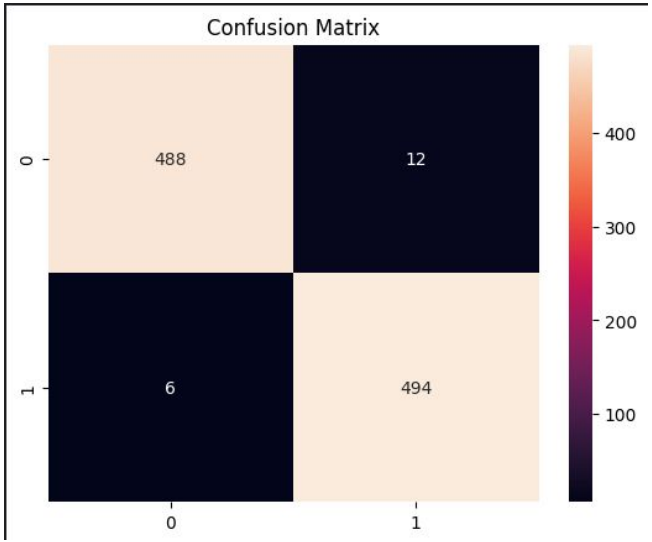

```python
# ---------- MODEL ----------
class PaSSTClassifier(nn.Module):
    def __init__(self):
        super().__init__()
        self.backbone = get_basic_model(mode="logits")
        self.backbone.net = get_model_passt(arch="passt_s_swa_p16_128_ap476", n_classes=512)  # Use default output size
        self.mlp = nn.Sequential(
            nn.Linear(512, 256),
            nn.ReLU(),
            nn.Dropout(0.3),
            nn.Linear(256, 64),
            nn.ReLU(),
            nn.Dropout(0.3),
            nn.Linear(64,2),
        )

    def forward(self, x):
        x, _ = self.backbone.net(x)
        x = self.mlp(x)
        return x
```

PaSST Results on Validation Data

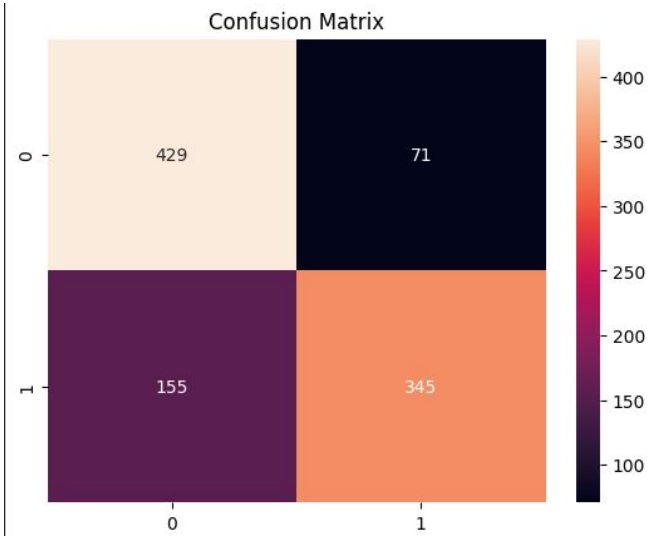


Confusion matrix for the Model trained on Full Dataset (PaSST + MLP)
Testing Accuracy: 99%
F1 Score : 99%

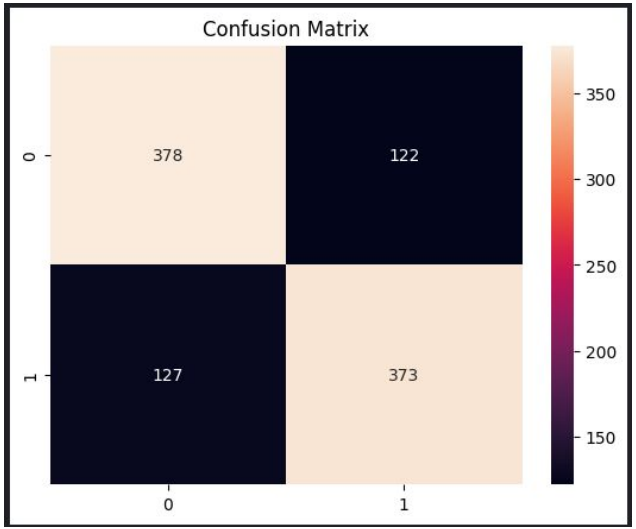


Confusion matrix for the Model trained on 10000 samples (PaSST + MLP)
Testing Accuracy : 99%
F1 Score : 98%

PaSST Results on Test Data



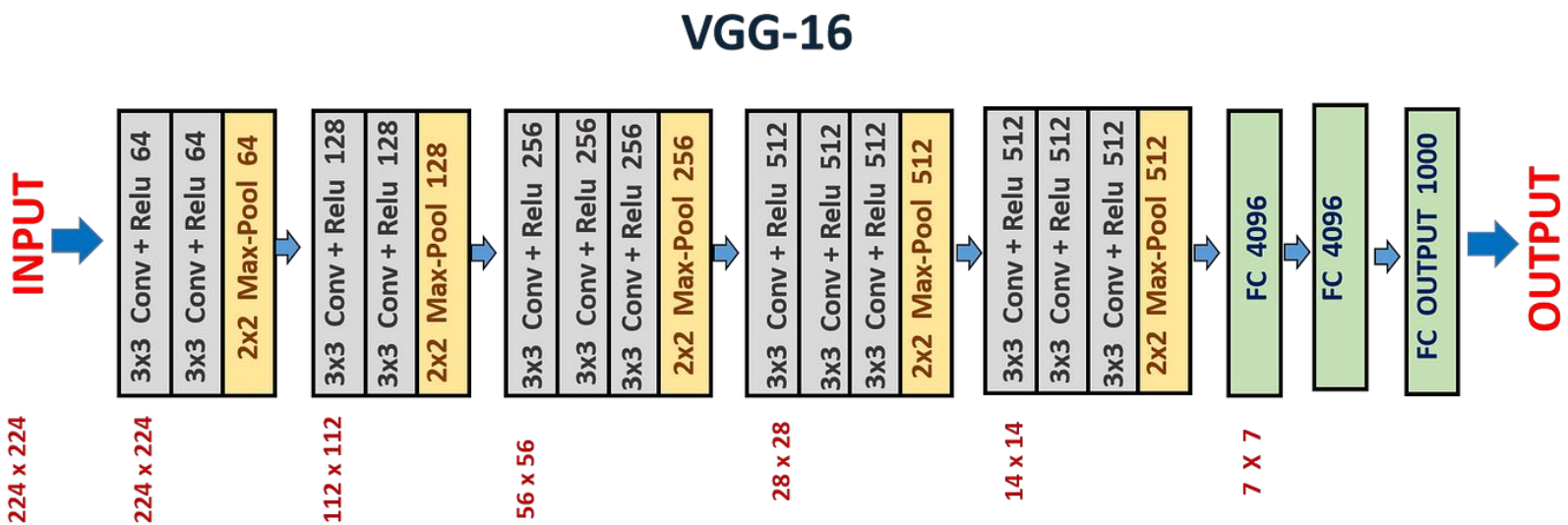
Confusion matrix for the Model trained on Full Dataset (PaSST + MLP)
Testing Accuracy: 77%
F1 Score : 77%



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

METHODOLOGY

- **VGG16 architecture**—originally designed for image classification—for audio deepfake detection by applying it to **MFCC (Mel-Frequency Cepstral Coefficients)** representations of speech. MFCCs effectively capture the timbral and phonetic characteristics of audio signals. By feeding MFCCs into VGG16, the model learns high-level acoustic features.



VGG 16 Results

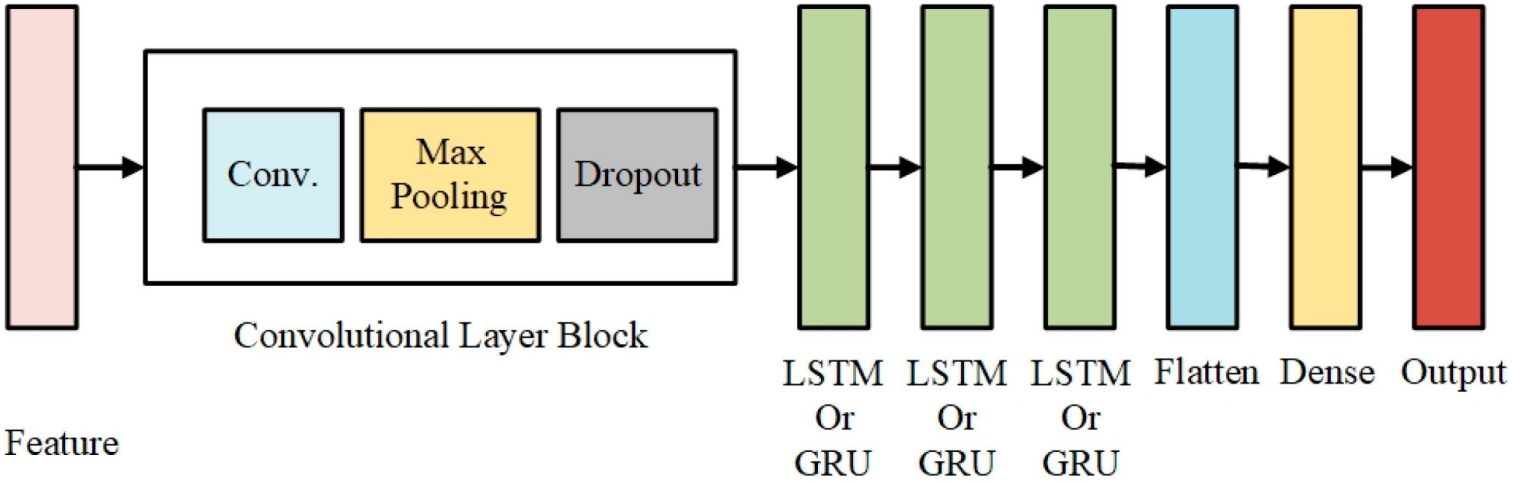
```
Training VGG16 model...
I0000 00:00:1746327492.337532      31 gpu_device.cc:2022] Created device /job:localhost/replica:0/task:0/device:GPU:0 with
pute capability: 6.0
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_k
58889256/58889256 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Epoch 1/20
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
I0000 00:00:1746327498.300322     104 service.cc:148] XLA service 0x7c338400fe50 initialized for platform CUDA (this does n
I0000 00:00:1746327498.300794     104 service.cc:156]   StreamExecutor device (0): Tesla P100-PCIE-16GB, Compute Capability
I0000 00:00:1746327498.694456     104 cuda_dnn.cc:529] Loaded cuDNN version 90300
  10/1684 ━━━━━━━━━━━━━━━━━━━━ 30s 18ms/step - accuracy: 0.7141 - loss: 0.6020
I0000 00:00:1746327503.044397     104 device_compiler.h:188] Compiled cluster using XLA!  This line is logged at most once
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.8888 - loss: 0.2797
Epoch 1: val_accuracy improved from -inf to 0.95092, saving model to norm_vgg16_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 47s 23ms/step - accuracy: 0.8888 - loss: 0.2796 - val_accuracy: 0.9509 - val_loss: 0.1276
Epoch 2/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9468 - loss: 0.1444
Epoch 2: val_accuracy improved from 0.95092 to 0.95962, saving model to norm_vgg16_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 33s 20ms/step - accuracy: 0.9468 - loss: 0.1444 - val_accuracy: 0.9596 - val_loss: 0.1051
Epoch 3/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9562 - loss: 0.1187
Epoch 3: val_accuracy improved from 0.95962 to 0.96990, saving model to norm_vgg16_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 33s 20ms/step - accuracy: 0.9562 - loss: 0.1187 - val_accuracy: 0.9699 - val_loss: 0.0902
Epoch 4/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9604 - loss: 0.1074
Epoch 4: val_accuracy improved from 0.96990 to 0.97120, saving model to norm_vgg16_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 33s 20ms/step - accuracy: 0.9604 - loss: 0.1074 - val_accuracy: 0.9712 - val_loss: 0.0817
Epoch 5/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9642 - loss: 0.0974
Epoch 5: val_accuracy improved from 0.97120 to 0.97240, saving model to norm_vgg16_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 33s 19ms/step - accuracy: 0.9642 - loss: 0.0974 - val_accuracy: 0.9724 - val_loss: 0.0736
Epoch 6/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9667 - loss: 0.0920
Epoch 6: val_accuracy did not improve from 0.97240
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 33s 19ms/step - accuracy: 0.9667 - loss: 0.0920 - val_accuracy: 0.9718 - val_loss: 0.0750
Epoch 7/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9699 - loss: 0.0798
Epoch 7: val_accuracy improved from 0.97240 to 0.97601, saving model to norm_vgg16_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 33s 20ms/step - accuracy: 0.9699 - loss: 0.0798 - val_accuracy: 0.9760 - val_loss: 0.0671
Epoch 8/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9708 - loss: 0.0797
Epoch 8: val_accuracy improved from 0.97601 to 0.97750, saving model to norm_vgg16_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 33s 20ms/step - accuracy: 0.9708 - loss: 0.0797 - val_accuracy: 0.9775 - val_loss: 0.0798
Epoch 9/20
1681/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9731 - loss: 0.0737
Epoch 9: val_accuracy did not improve from 0.97750
```

```
Final results:

Dataset: norm
  VGG16 model:
    accuracy: 0.8845
    precision: 0.8847
    recall: 0.8845
    f1: 0.8846
```

METHODOLOGY

- **Long Short-Term Memory (LSTM)** network to capture the **temporal dynamics** in audio signals. Unlike CNNs, which focus on spatial patterns, LSTMs excel at modeling **sequential dependencies**, making them well-suited for processing MFCC sequences over time. The LSTM model learns how acoustic features evolve throughout an utterance, enabling it to distinguish between real and fake speech based on voice consistency, cadence, and temporal patterns.



LSTM Results

```
Training LSTM model...
Epoch 1/20
1683/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - accuracy: 0.7926 - loss: 0.4404
Epoch 1: val_accuracy improved from -inf to 0.92378, saving model to norm_lstm_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 22s 10ms/step - accuracy: 0.7927 - loss: 0.4403 - val_accuracy: 0.9238 - val_loss: 0.2083
Epoch 2/20
1679/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - accuracy: 0.9335 - loss: 0.1921
Epoch 2: val_accuracy improved from 0.92378 to 0.95684, saving model to norm_lstm_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 17s 10ms/step - accuracy: 0.9335 - loss: 0.1921 - val_accuracy: 0.9568 - val_loss: 0.1344
Epoch 3/20
1683/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - accuracy: 0.9585 - loss: 0.1265
Epoch 3: val_accuracy improved from 0.95684 to 0.97018, saving model to norm_lstm_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 17s 10ms/step - accuracy: 0.9585 - loss: 0.1265 - val_accuracy: 0.9702 - val_loss: 0.0990
Epoch 4/20
1680/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - accuracy: 0.9728 - loss: 0.0806
Epoch 4: val_accuracy improved from 0.97018 to 0.97398, saving model to norm_lstm_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 17s 10ms/step - accuracy: 0.9728 - loss: 0.0806 - val_accuracy: 0.9740 - val_loss: 0.0758
Epoch 5/20
1683/1684 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step - accuracy: 0.9781 - loss: 0.0665
Epoch 5: val_accuracy improved from 0.97398 to 0.97833, saving model to norm_lstm_model.keras
1684/1684 ━━━━━━━━━━━━━━━━━━━━ 17s 10ms/step - accuracy: 0.9781 - loss: 0.0665 - val_accuracy: 0.9783 - val_loss: 0.0803
145/145 ━━━━━━━━━━━━━━━━━━━━ 3s 18ms/step - accuracy: 0.8909 - loss: 0.2778
145/145 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - accuracy: 0.7178 - loss: 0.6241
145/145 ━━━━━━━━━━━━━━━━━━━━ 3s 16ms/step
145/145 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step
```

```
LSTM model:
  accuracy: 0.8259
  precision: 0.8600
  recall: 0.8259
  f1: 0.8209
```

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

Baseline Models and Accuracy

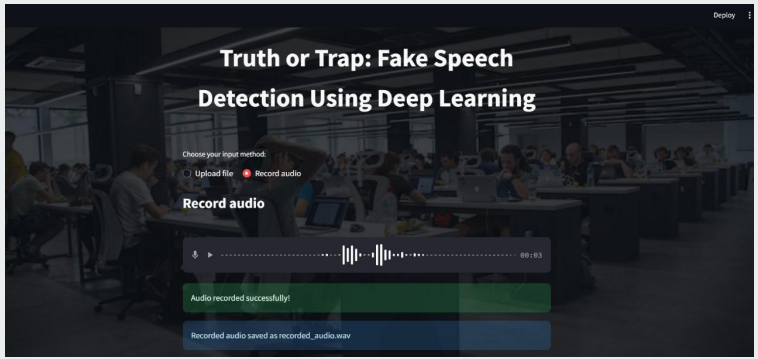
Ensemble Learning Results

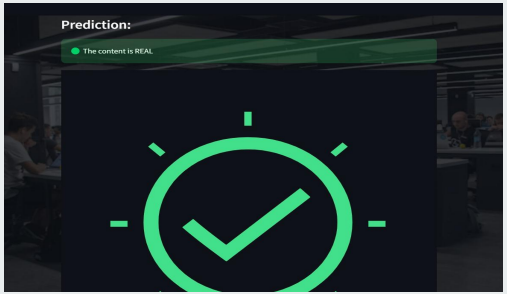
To enhance classification performance, we implement a weighted ensemble strategy that combines the strengths of three models: VGG16, PaSST, and LSTM. Each model contributes to the final prediction based on its individual performance, with weights assigned as follows: VGG16 – 0.40, PaSST – 0.25, and LSTM – 0.35. This weighting average reflects their comparative accuracies and ensures that both spatial and temporal features are effectively captured. The ensemble model demonstrates improved overall accuracy and robustness, especially in cases where individual models show uncertainty or disagreement.

The results of the ensemble learning are as follows:

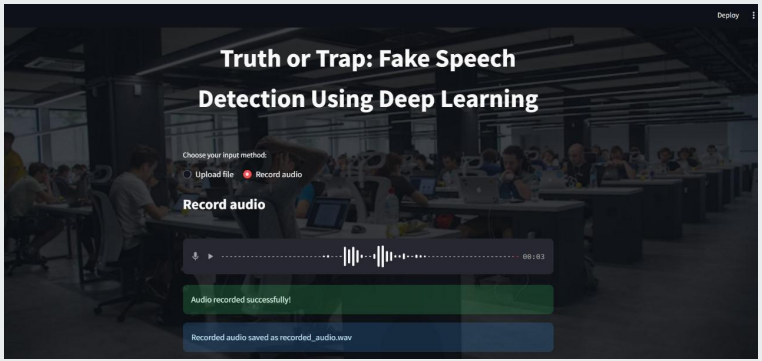
Accuracy : 0.9102
Precision : 0.9024
Recall : 0.8991
F1 : 0.9002

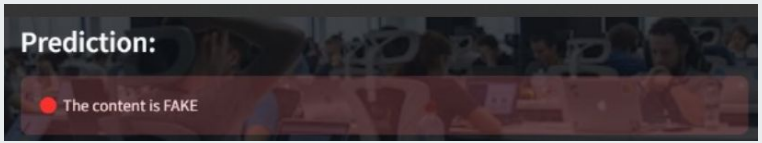
Web Application Results





REAL SAMPLE PROVIDED





FAKE SAMPLE PROVIDED

Discussion and Conclusions

- Diverse Feature Learning: By combining PaSST (spectral patterns), VGG16 (spatial MFCC features), and LSTM (temporal dependencies), we leverage complementary strengths for a more accurate and comprehensive audio classification.
- Both real voice frequencies showed time-frequency similarities with synthetic voice frequencies at intermittent points.
- The sophisticated capabilities of PaSST required extensive driving force usage and optimization for maximizing its operating speed.
- Some fake examples included background noise to create noises commensurate with authentic samples.

References

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

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

[3] D. Cai, X. Xu, and M. Wang, “Audio Scene Classification Using VGG and ResNet Neural Networks,” in *2019 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW)*, 2019

[4] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, “PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020

Thank you