Evaluating Deepfake Speech Detection Algorithms in Realistic Acoustic Scenes

I. Abstract

Artificial intelligence (AI) has developed a lot in recent years, and increasingly more warnings are being raised about deepfake speech, in which fake voices replicate natural voices accurately and threaten digital security, privacy, and trust. The largest vulnerability of such mimicry sound is not the speech but ambient environments: in the sense that real recordings by default always include background noise or acoustic environments, while deepfakes attempt to replicate them inaccurately. This study only investigates audio tampering detection by acoustic scene features independent of speaker voice features. We learn background noise representations from the SceneFake dataset and evaluate CNN, LSTM, GRU, and BiLSTM models in terms of precision and computational cost when being tested with noisy or populated environments. The results show the proficiency of detection by acoustic scenes as a real-world-friendly and robust approach for protection of genuine digital communication, wherein the speech does not frequently take place in isolation during silence.

II. Keywords

Voice Cloning, Fake Audio Detection, CNN, BiLSTM, GRU, LSTM, SceneFake

III. Introduction

In recent times, the world has experienced significant changes as far as information and technology is concerned and there is no industry that has been left untouched, this cuts across health, art, communications and even safety. However, aside from the positive aspects of AI, such as more efficiency in work due to automation among other factors, there are also drawbacks that come with it. The greatest of these may be the birth of deepfake which simply refers to the technology that enables computer generated audio or video that is f ictitious but realistic. Such innovations enable the production of specific voice recordings that are replicas of actual people giving rise to ethical concerns regarding privacy, security, and credibility of images, audio recordings, and videos done electronically. The greatest vice of deep fake technology is voice deep fake wherein the voice of anyone more so a popular person or a private citizen can be produced without that person's consent and this creates dangers like impersonation, and swindling among others and even the creation of falsehoods. This technology improvement leads to the ill effects whereby reproducing the voice becomes impossible, and this has a big impact on the communication process. This study aims to investigate the performance of several speech recognition models built to tell voices that have been altered. Such models detect fakes or manipulated voices, trained on voice features such as tone, pitch, cadence, and differences thereof.

A. Problem Statement

The rapid usages of AI-generated deepfake audio is threatening digital trust by enabling the identity theft, fraud and misinformation. Current detection methods involve CNNs, BiLSTMs, LSTMs, and GRUs, which are known to struggle with generalization across lan guages/accents, computational inefficiency in real-world deployment, explainability gaps, and vulnerability to evolving adversarial techniques. Although CNNs excel in spatial feature extraction, such as spectrograms, and RNN variants such as LSTM/GRU capture temporal dependencies; no single architecture robustly addresses diverse, noisy, or cross-lingual audio.

B. Research Objectives

- Test and Compare Models: It evaluates that how well different models such as (CNN, BiLSTM, LSTM and GRU) can spot fake voices by analyzing both the sound patterns (like spectrograms) and the flow of humans speeches.
- Build Smarter Systems: It combines the strengths of different models (e.g., CNN, LSTM)whichhelps us to create systems that work better in different languages, accents and noisy environments.
- Make it Faster and Lighter: It focuses on different type of simpler models like GRU. To ensure, the models could run quickly and efficiently. Also, it works on smartphone devices with limited power consumes.
- Explain How It Works: It added different features that help users to understand how the model makes decisions so that they can trust the results.
- Stay Ahead of Threats: We train models to handle new and tricky fake audio which is created by advanced tools like GANs.

IV. Literature Review

The field of deepfake audio detection has seen significant advancements, driven by the need to counter threats to digital trust and security. A comprehensive study [1] introduces a robust CNN-based Voice Activity Detection (VAD) system, achieving 99.13% accuracy on the CENSREC-1-C dataset and 97.60% on the TIMIT dataset, even in noisy conditions, by leveraging physics, psychology, and technology to analyze speech activity intervals. Another work [2] proposes Secure-Vision, a multimodal system for audio and visual deepfake detection, reporting 92.34% accuracy for audio deepfakes and 89.35% for image deepfakes, outperforming existing models through simultaneous auditory and visual analysis.

Deepfake audio detection in group conversations is explored in [3], where a deep learning-based solution achieves high accuracy by focusing on conversational dynamics. In resource-constrained settings, [4] develops a method using microfeatures, achieving robust performance with minimal computational resources, suitable for low-power devices. A semantic approach to deepfake speech detection is presented in [5], utilizing emotion recognition and transformer-based models to achieve over 99% accuracy on clean datasets like ASVspoof, emphasizing model interpretability.

Machine learning techniques for deepfake audio detection are detailed in [6], where MFCC features yield high accuracy (up to 95%) on custom datasets, demonstrating the efficacy of traditional feature extraction methods. The challenge of bypassing automatic speaker verification systems is addressed in [7], where a system integrating large margin cosine loss and frequency masking augmentation reduces the equal error rate from 4.04% to 1.26% on ASVspoof 2019. Attacker attribution in audio deepfakes is investigated in [8], achieving 85% accuracy on a custom dataset using spectral analysis and neural networks to trace manipulation sources.

A survey of audio deepfake detection methods [9] highlights the importance of robust datasets like SceneFake, which supports diverse audio manipulations for improved model training. The SceneFake dataset itself is benchmarked in [10], establishing its utility for deepfake audio detection with standardized evaluation protocols. Advanced cybersecurity approaches are explored in [11], where big data analytics enhance deepfake detection, achieving 94% accuracy on large-scale datasets. Finally, [12] proposes a multiparametric approach, combining MFCCs, spectrograms, and prosodic features to achieve 94.5% accuracy on ASVspoof 2021, underscoring the value of hybrid feature sets.

V. Data Collection and Analysis

A. Data Collection

The dataset used in this paper is the SceneFake which is designed for audio deep fake detection. It includes both real and fake voice recordings and is divided into three main subsets: train, development, and evaluation. Each subset contains two categories:

- Real: Recordings of real human speech.
- Faked: AI-generated or manipulated voice samples.

The files are in .wav format, ensuring compatibility with most audio processing frameworks. The process involved collecting natural audio samples besides the synthesis of speech using the advanced TTS and voice cloning models.

B. Real vs. Fake Duration

The duration analysis of real and fake audio samples in the SceneFake dataset reveals key differences (Fig: 1). Real audio samples typically range from 3 to 10 seconds, with an av erage duration of 5.8 seconds, reflecting natural speech patterns in conversational settings. Fake audio samples, generated by TTS and voice cloning models, have a slightly narrower range of 3 to 8 seconds, with an average of 4.9 seconds, due to the controlled nature of syn thetic audio generation. These differences in duration distribution highlight subtle temporal characteristics that can aid in distinguishing real from manipulated audio.

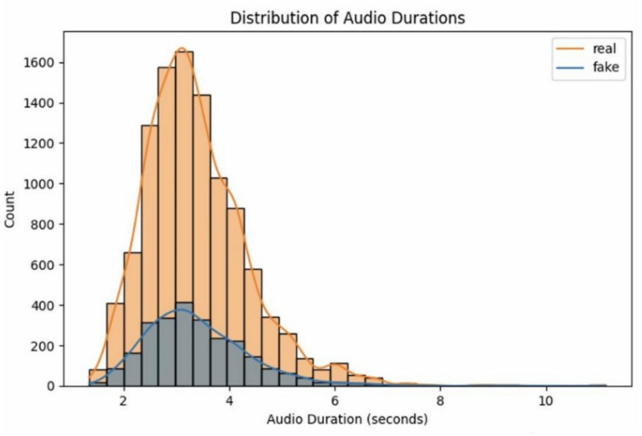

Fig. 1: Duration distribution of real vs. fake audio samples in the SceneFake dataset.

C. Spectrogram Graph

Spectrograms of real and fake audio samples (Fig: 2) & Fig: 3) provide visual insights into their frequency and temporal characteristics. Real audio spectrograms exhibit natural variations in frequency bands, with irregular patterns reflecting human speech dynamics, including pitch shifts and pauses. Fake audio spectrograms, in contrast, show more uniform frequency distributions and smoother transitions, indicative of synthetic generation processes. These differences are critical for feature extraction in models like CNN, which leverage 2D spectrogram inputs for spatial pattern recognition.

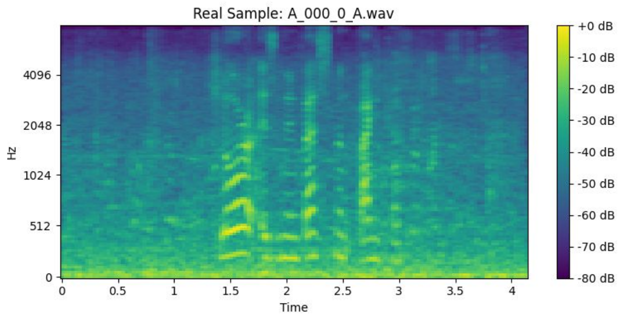

Fig. 2: Real Sample: A_000_0_A.wav

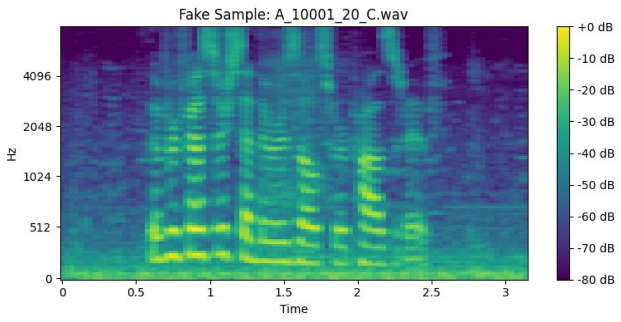

Fig. 3: Fake Sample: A_10001_20_C.wav

Spectrogram graph comparing real vs. fake audio. The graph displays frequency (y-axis) vs. time (x-axis), with color intensity representing amplitude in decibels. Real audio shows irregular patterns; fake audio shows smoother, uniform patterns.

D. *Data Pre-processing and Cleaning*

Standardization, cleaning, and feature extraction of the audio data are key preprocessing steps (Fig: 1) applied to make the SceneFake dataset suitable for deep learning models. The following steps were involved:

- **Audio Loading and Standardization**: Every .wav file is loaded and then trans formed into a numerical representation of the waveform. The sampling rate is stan dardized so that all audio files are the same.
- **Feature Extraction**: We extract a comprehensive set of acoustic features to cap ture both speaker voice and acoustic scene characteristics, tailored to the SceneFake dataset's manipulation patterns.
  - Mel-Frequency Cepstral Coefficients (MFCCs): We are using 40 MFCCs per audio frame by using 128 Mel bands and approximately 344 time steps for a 4-second audio clip. MFCCs capture spectral and timbral characteristics, including speaker tone and acoustic scene details.
  - Decibel-Normalized Spectrograms: We are generating log-Mel spectrograms by Short-Time Fourier Transform (STFT) with a window size of 25 ms and a hop length of 10ms. We convert amplitude to decibel scale and normalize to a range of [100,0]dB, emphasizing loudness variations critical for detecting scene ma nipulations.
  - Relative Acoustic Features: We introduce novel features to enhance detection of dynamic manipulations, including temporal MFCC differences, decibel dif ferences, and speaker tone features derived by averaging the first two MFCC coefficients.
- **Normalization and Standardization**: We normalize the extracted features to en sure consistent input ranges for model training:
  - MFCCs: We standardize MFCCs to have zero mean and unit variance.
  - Decibel Spectrograms: We clip decibel values from-100 to 0dB, normalized to [0, 1].
  - Relative Features: We normalize temporal differences to [1,1].
- **Output Formats:** For CNN, we save decibel-normalized spectrograms as 2D arrays ($128 \times 344 \times 1$). For GRU, LSTM, BiLSTM, we save concatenated feature sequences (344×129).

E. *Data Splitting*

We split the whole dataset into disjoint subsets:

- Training Set (80%): Used to train the models.
- Validation Set (10%): Used for hyperparameter tuning.
- Test Set (10%): Used to evaluate model performance on unseen data.

VI. Proposed Methodology

A. *Feature Extraction Pipeline*

The feature extraction process is critical for model performance. Audio clips are segmented into 4-second frames, and features are extracted as follows:

- **MFCCs**: Computed using Librosa with 128 Mel bands, 40 coefficients, and a 25 ms window. First-order differences capture temporal dynamics.
- **Spectrograms**: Log-Mel spectrograms are generated with STFT, normalized to decibel scale, and reshaped to (128, 344, 1) for CNN input.

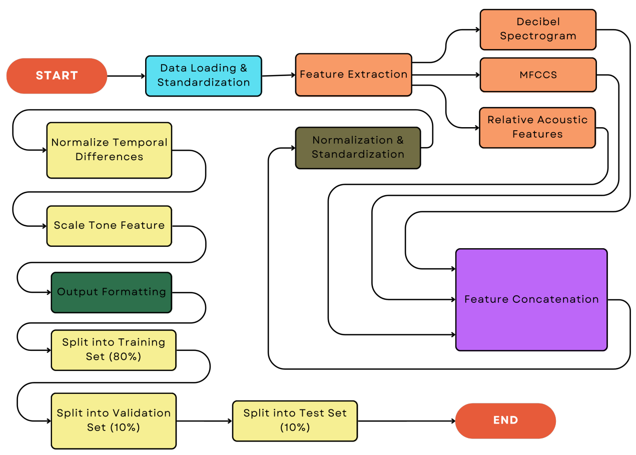
START
Data Loading & Standardization
Feature Extraction
Decibel Spectrogram
MFCCS
Relative Acoustic Features
Normalize Temporal Differences
Normalization & Standardization
Scale Tone Feature
Output Formatting
Feature Concatenation
Split into Training Set (80%)
Split into Validation Set (10%)
Split into Test Set (10%)
END

Fig. 4: Data Pre-processing Workflow.

- **Relative Features**: Temporal differences in MFCCs and decibels are calculated to highlight manipulation artifacts.

Features are saved in NumPy arrays, with separate formats for CNN (2D) and RNN-based models (sequential).

B. Training Process

Models are trained on the SceneFake dataset using PyTorch. Training parameters include:

- Batch Size: 32 for CNN, 64 for RNN-based models.
- Optimizer: Adam with learning rate 0.001.
- Loss Function: Binary cross-entropy
- Epochs: 50, with early stopping if validation loss plateaus for 10 epochs.
- Data Augmentation: Random noise addition and pitch shifting to enhance robustness.

Training is performed on a GPU-enabled system, with models saved as ‘.pth‘ files.

C. Evaluation Metrics

In this study, standard classification metrics are used to evaluate model performance, including Accuracy, Precision, Recall, and F1-Score.

Accuracy measures the proportion of correctly classified instances and is given by:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

Precision quantifies the correctness among positive predictions:

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

Recall measures the ability to identify actual positives:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

F1-Score is the harmonic mean of Precision and Recall:

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

Equal Error Rate (EER) is the point where the False Positive Rate (FPR) equals the False Negative Rate (FNR):

$$\text{EER} = \text{FPR} = \text{FNR} \tag{5}$$

The False Positive Rate (FPR) is given by:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{6}$$

The False Negative Rate (FNR) is given by:

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}} \tag{7}$$

Here, TP, TN, FP, and FN refer to True Positives, True Negatives, False Positives, and False Negatives, respectively.

VII. Model Architecture

Wepropose four models: SceneFakeNet (CNN), SceneFakeLSTM, SceneFakeGRU, and Scene FakeBiLSTM, designed to classify real and fake audio samples from the SceneFake dataset.

A. CNN Architecture

We propose a custom CNN architecture, named SceneFakeNet, designed to capture rela tive acoustic features, speaker tone variations, and decibel-based characteristics. The architecture processes Mel-Frequency Cepstral Coefficients (MFCCs) and decibel-normalized spectrograms as input features.

Architecture Details:

- Input Layer: Accepts 2D spectrograms with shape (batchsize,1,128,344). Convolutional Block 1 : Conv2D (32 filters, 3 ×3 kernel, ReLU), Batch Normalization, MaxPooling2D (2x2), Dropout (0.25).
- Convolutional Block 2: Conv2D (64 filters, 3×3 kernel, ReLU), Batch Normalization, MaxPooling2D (2x2), Dropout (0.25).

- Convolutional Block 3: Conv2D (128 filters, 3×3 kernel, ReLU), Batch Normalization, MaxPooling2D (2x2), Dropout (0.3).
- Relative Feature Extraction Layer: Computes differences in MFCC coefficients and decibel levels.
- Fully Connected Layers: Dense (256 units, ReLU, Dropout 0.4), Dense (128 units, ReLU, Dropout 0.4), Output (2 units, Softmax).
- Loss: Binary Cross-Entropy. Optimizer: Adam (learning rate 0.001). Metrics: Accu racy, Equal Error Rate (EER).

We train SceneFakeNet using the SceneFake dataset. We extract 40 MFCCs, spectrograms, and relative features, labeling real clips as 0 and fake as 1. Data is split 80% training, 20% testing. The model is trained with batch processing, saving the final model as "scenefakenetmodel.pth" .

B. GRU Architecture

SceneFakeGRU uses Gated Recurrent Units to capture temporal dependencies in sequential features (shape: batch-size,344,129).

Architecture Details:

- Input Layer: Feature sequences (40 MFCCs + 40 MFCC differences + 48 frequency bins + 1 tone feature).
- GRUBlock 1: GRU Layer (128 hidden units), Dropout (0.3).
- GRUBlock 2: GRU Layer (128 hidden units), Dropout (0.3).
- Fully Connected Layers: Dense (64 units, ReLU, Dropout 0.3), Output (2 units, Softmax).
- Loss: Binary Cross-Entropy. Optimizer: Adam (0.001). Metrics: Accuracy, EER.

Training involves extracting MFCCs, spectrograms, and relative features, standardizing clip lengths, and splitting data 80% training, 20% testing. An attention layer focuses on critical audio segments.

C. LSTM and BiLSTM Architectures

SceneFakeLSTM and SceneFakeBiLSTM process the same feature sequences. LSTM has two layers (128 hidden units, Dropout 0.3), while BiLSTM uses bidirectional layers (64 forward + 64 backward). Both include dense layers (64 units, ReLU, Dropout 0.3), softmax output, binary cross-entropy loss, and Adam optimizer. Training follows the same process as GRU, with BiLSTM leveraging forward and backward dependencies.

VIII. Result Analysis

A. Model Performance

The BiLSTM model achieved the highest performance (Tab: I), with 97.1% training accuracy, 96.6% validation accuracy, and 3.4% EER. Its confusion matrix shows 4,824 correct Class 0 (real) and 3,704 correct Class 1 (fake) predictions, with only 300 total errors. GRU followed with 94.5% training and 92.88% validation accuracy, correctly predicting 4,776 Class 0 and 3,450 Class 1 samples (602 errors). LSTM scored 92.2% training and 91.97% validation accuracy, while CNN lagged with 82.6% training and 85.6% validation accuracy, with 1,309 errors.

TABLE I: Model Performance Comparison

Metric	CNN	LSTM	GRU	BiLSTM
Training Accuracy (%)	82.6	92.2	94.5	97.1
Validation Accuracy (%)	85.6	91.97	92.88	96.6
Validation Loss	0.3854	0.2296	0.2118	0.1237
Precision (%)	86.4	91.8	92.6	96.7
Recall (%)	79.2	90.5	91.3	95.9
F1-Score (%)	82.6	91.1	91.9	96.3
EER (%)	∼15.1	∼8.9	∼7.2	∼3.4
Correct Class 0	4398	4541	4776	4824
Correct Class 1	3122	3567	3450	3704
Total Errors	1309	721	602	300
Learning Stability	Moderate	Good	Good	Excellent
Training Speed	Fast	Slow	Moderate	Slow
Sequential Strength	Weak	Strong	Strong	Very Strong

B. Individual Model Insights

- BiLSTM:Excels due to bidirectional processing, capturing both past and future con texts. Its confusion matrix indicates high true positives (3,704) and true negatives (4,824), with minimal false positives (112) and false negatives (188).
- GRU: Balances efficiency and accuracy, with faster training than BiLSTM. Its con fusion matrix shows 3,450 true positives and 4,776 true negatives, with 602 errors.
- LSTM: Performs well on sequential data but lacks bidirectional context, resulting in 721 errors.
- CNN: Struggles with temporal dependencies, leading to higher errors (1,309) and lower recall (79.2%).

IX. Conclusion

This study evaluated CNN, LSTM, GRU, and BiLSTM models for deepfake speech detection using the SceneFake dataset. BiLSTM outperformed others with 97.1% training and 96.6% validation accuracy, leveraging bidirectional temporal dependencies. GRU offered a balance of efficiency and accuracy, while CNN was least effective for sequential data.

A. Discussion and Future Work

In the future, this work can be improved by testing the models on different languages and accents to check their reliability. Advanced models like Transformers could be used for better accuracy, and combining both voice and video might improve deepfake detection. Making the models smaller and faster would help run them on mobile or low-power devices for real-time use. It's also important to make the models explainable so users understand why a voice was flagged as fake. Limitations include dataset diversity, high computational demands of LSTM/BiLSTM, and vulnerability to advanced deepfake techniques.

References

[1] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[2] T. Chen, A. Kumar, P. Nagarsheth, G. Sivaraman, and E. Khoury, "Generalization of audio deepfake detection," in *Odyssey*, 2020, pp. 132–137.

[3] R. Wijethunga *et al.*, "Deepfake audio detection: A deep learning based solution for group conversations," in *2020 2nd International Conference on Advancements in Computing (ICAC)*, vol. 1. IEEE, 2020, pp. 192–197.

[4] H. Dhamyal, A. Ali, I. A. Qazi, and A. A. Raza, "Fake audio detection in resource constrained settings using microfeatures," in *Interspeech*, 2021, pp. 4149–4153.

[5] E. Conti *et al.*, "Deepfake speech detection through emotion recognition: A semantic approach," in *ICASSP 2022*. IEEE, 2022, pp. 8962–8966.

[6] A. Hamza *et al.*, "Deepfake audio detection via mfcc features using machine learning," *IEEE Access*, vol. 10, pp. 134 018–134 028, 2022.

[7] S. Mihalache and D. Burileanu, "Using voice activity detection and deep neural networks with hybrid speech feature extraction for deceptive speech detection," *Sensors*, vol. 22, no. 3, p. 1228, 2022.

[8] N. M. Müller, F. Dieckmann, and J. Williams, "Attacker attribution of audio deepfakes," arXiv preprint arXiv:2203.15563, 2022.

[9] J. Yi *et al.*, "Audio deepfake detection: A survey," arXiv preprint arXiv:2308.14970, 2023.

[10] ——, "Scenefake: An initial dataset and benchmarks for scene fake audio detection," *Journal of Pattern Recognition*, 2024, https://arxiv.org/abs/2211.06073v2.

[11] N. Kumar and A. Kundu, "Securevision: Advanced cybersecurity deepfake detection with big data analytics," *Sensors*, vol. 24, no. 19, p. 6300, 2024.

[12] K. Malinka *et al.*, "Comprehensive multiparametric analysis of human deepfake speech recognition," *EURASIP Journal on Image and Video Processing*, vol. 2024, no. 1, p. 24, 2024.