iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VI **Month of publication:** June 2026

DOI: https://doi.org/10.22214/ijraset.2026.83345

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VI June 2026- Available at www.ijraset.com

AI-Based Deepfake Audio Detection with Noise-Aware Spectrogram Processing

Prasadbabu Karri[1], B.Jhansi[2], Y. Jagadeesh Kumar[3]
[1, 2, 3]*Sri Sivani College Of Engineering, India*

Abstract: ***Deepfake audio has emerged as a significant cyber security and social threat due to rapid advancements in artificial intelligence and speech synthesis technologies. Fraudulent audio generated using deep learning techniques can imitate human voices with high accuracy, making the identification of fake speech increasingly difficult, especially in noisy real-world environments. To address this challenge, this work proposes an AI-Based Deepfake Audio Detection with Noise-Aware Spectrogram Processing that enhances detection reliability under varying acoustic conditions.***

The proposed model utilizes advanced audio preprocessing techniques to remove background noise and convert speech signals into spectrogram representations, enabling efficient extraction of temporal and frequency-domain features. A deep learning framework based on Convolutional Neural Networks (CNN) is employed to automatically learn discriminative patterns between genuine and manipulated audio samples. Noise-aware feature enhancement is integrated into the preprocessing stage to improve robustness against environmental disturbances and signal degradation.

The system is trained and evaluated on benchmark deepfake audio datasets containing both authentic and AI-generated speech samples with multiple noise levels. Experimental results demonstrate improved classification accuracy, reduced false detection rates, and enhanced generalization performance compared to conventional deepfake detection methods. The proposed framework can be effectively applied in voice authentication, digital forensics, cybersecurity, media verification, and fraud prevention systems.

This research contributes to the development of reliable and intelligent audio forensic solutions capable of detecting sophisticated deepfake attacks in realistic noisy environments.

Index Terms—[Font: Times New Roman, Size:9]Aboutfour(min

I. INTRODUCTION

The rapid advancement of artificial intelligence and deep learning technologies has fun- damentally transformed the way speech is synthesized and processed. Voice cloning, text-to-speech (TTS) synthesis, and voice conversion systems have reached a level of realism that makes it increasingly difficult to distinguish between genuine human speech and machine-generated audio. While these technologies offer significant benefits in ar- eas such as virtual assistants, automated dubbing, and accessibility tools for persons with disabilities, they also present a growing and serious threat to digital security, iden- tity verification, and public trust. Deepfake audio refers to synthetic or artificially generated speech that closely mimics a real human voice. These audio files can be generated with minimal data and computa- tional resources using modern deep learning architectures. The rise of open-source voice synthesis platforms has further democratized the creation of deepfake audio, making it accessible not only to researchers but also to malicious actors. Instances of deepfake audio misuse include impersonation fraud in banking and financial institutions, the cre- ation of fake emergency calls, manipulation of political discourse through fabricated voice recordings, and unauthorized access to voice-based biometric authentication sys- tems.

In response to these emerging threats, deepfake audio detection has become a critical area of research in digital forensics, cybersecurity, and speech processing. Detection systems aim to identify acoustic anomalies and artifacts introduced during the speech synthesis process that are not present in authentic human speech. These systems are typically trained on benchmark datasets and evaluated under controlled, clean recording conditions, yielding high detection accuracy in laboratory environments. However, the performance of these systems degrades substantially when deployed in real-world scenarios. Real-world audio environments introduce a range of acoustic chal- lenges, including background noise from traffic, crowded spaces, or machinery; channel distortions from telephony systems or compression codecs; reverberation effects from different room acoustics; and variations in recording quality across different microphone types. Most existing detection systems have not been designed or trained to handle such variability, and their robustness in noisy conditions remains limited.

©IJRASET: All Rights are Reserved

Furthermore, many real-world applications such as call center fraud detection, large- scale voice authentication systems, and social media content moderation require the analysis of multiple audio streams simultaneously. Existing deepfake audio detection frameworks, which are typically designed to process a single audio file at a time, are ill- suited for such high-throughput deployments. This scalability limitation significantly restricts the practical utility of current detection approaches. This project presents a noise-aware deepfake audio detection system designed to address these critical gaps. The proposed system leverages spectrogram-based audio feature rep- resentations, specifically Mel-Frequency Cepstral Coefficients (MFCCs) and Mel-spec- trograms, combined with a hybrid Convolutional Neural Network – Bidirectional Long Short-Term Memory (CNN-BiLSTM) deep learning architecture. Noise-aware data augmentation techniques are employed during training to improve the model's robust- ness to diverse real-world acoustic conditions. Additionally, the system incorporates batch-based parallel inference to support the simultaneous processing of multiple audio inputs, making it suitable for deployment at scale.

The system is developed and evaluated using the ASVspoof 2019 Logical Access (LA) dataset, supplemented with real-world noise samples from the MUSAN and Ur- banSound8K datasets. Experimental results demonstrate that the proposed system achieves reliable detection accuracy across both clean and noisy audio conditions while maintaining computational efficiency through parallel processing.

II. LITERATURE SURVEY

The Noise-Aware Deepfake Audio Detection via Spectrogram Based Deep Neural Networks project focuses on existing deepfake audio detection techniques and their limitations. Previ- ous studies have mainly used machine learning and deep learning models such as CNN, RNN, and LSTM to identify fake audio by analyzing speech features like MFCC and spec- trograms. Many works have shown good accuracy in controlled environments, but most of them struggle in real-world conditions where background noise and multiple audio sources are present. Some recent approaches have attempted to improve performance using advanced models and larger datasets, but challenges like noise robustness and scalability still remain. Based on these observations, this project aims to improve detection by introducing noise- aware training and a hybrid CNN + BiLSTM model to achieve better accuracy in practical scenarios.

Muhammad Owais developed a deepfake audio detection framework for low-resource lan- guages, specifically Urdu, to address challenges in digital security and media forensics. They evaluated multiple deep learning models, including LCNN, CNN–LSTM with Attention, and transformer-based Whisper variants, using MFCC and Linear Frequency Cepstral Coef- ficients for feature extraction. To overcome data limitations, they created baseline, aug- mented, and extended datasets through controlled augmentation and additional recordings. Their experiments showed that MFCC-based models, particularly Whisper-small, achieved strong performance with an Equal Error Rate as low as 0.50%. The study also highlighted that transformer-based models are more robust than lightweight CNNs under noise, pitch, and tempo variations. This work provides a structured benchmark for Urdu deepfake detec- tion and emphasizes the importance of multilingual evaluation for building reliable speech forensics systems.[1]

KAVYA SREE KAMMARI proposed an rPPG-based deepfake detection approach that an- alyzes subtle skin color variations to extract physiological signals such as heart rate for iden- tifying fake videos. The method combines rPPG signal enhancement techniques with ma- chine learning models to improve detection accuracy. It is applied in real-world scenarios, including detecting manipulated celebrity videos used in propaganda. The study highlights that this approach effectively detects deepfakes by capturing physiological inconsistencies that are difficult to replicate, though performance may vary depending on video quality and environmental conditions, providing a promising solution for secure digital media verifica- tion.[2]

DAEUN SONG proposes an unsupervised learning approach to detect deepfake audio by learning patterns of real human voices instead of relying on labeled fake data. It converts audio into features like Mel-Spectrogram and MFCC, then uses GAN-based models such as GANomaly to identify anomalies in speech. The system assigns an anomaly score to deter- mine whether the audio is real or fake. The results show strong performance, achieving an F1-score of 93%, making it effective for detecting new types of deepfake audio without prior training on fake samples.[3]

OUSAMA A. SHAABAN proposes different techniques used for creating and detecting deepfake audio, focusing on both traditional and advanced methods. It discusses approaches like statistical analysis, media consistency, and machine learning models such as SVM, CNN, and RNN. The study compares multiple algorithms and shows that performance varies depending on the method usedand Decision Trees the lowest at 73.33%. It also highlights evaluation metrics like EER and t-DCF, where advanced models like Siamese CNN perform better. Overall, the review provides a clear understanding of existing detection techniques and their effectiveness in identifying deepfake audio.[4]

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

GHULAM ALI proposes a deep ensemble learning approach called ECN-MF for detecting deepfake audio by combining multiple neural network models like CNN, RNN, LSTM, and ConvLSTM. It extracts various audio features such as MFCC, chroma, and zero-crossing rate and applies preprocessing techniques to improve data quality. The model was tested on different versions of the Fake-or-Real. The results show that combining multiple models improves performance and makes the system more effective than using a single model for deepfake audio detection.[5]

DEMAO XIONG proposes a deep learning model called BMNet to detect forged videos by capturing both spatial and temporal features. It uses facial landmark data from video frames and applies BiLSTM to understand changes over time.The model was tested on multiple datasets and showed improved performance compared to traditional CNN-based methods, achieving high accuracy across different datasets. This approach effectively detects both dy- namic and localized forgery features, making it a strong solution for deepfake video detec- tion.[6]

KAVYA VERMA explores advanced deep learning models for detecting deepfake audio by extracting both spectral and temporal features from speech signals. Various models such as BLSTM, CNN, and hybrid architectures with attention mechanisms were compared to eval- uate their performance. The results showed that the SE-Enhanced 1D-CNN model achieved the highest accuracy of 97.64%, demonstrating strong detection capability. Additionally, the study found that using 39 MFCC features provided the best representation for identifying fake audio. Overall, the work highlights the effectiveness of combining multiple deep learn- ing techniques and feature types to improve deepfake audio detection accuracy.[7]

DIVYANSHI MITTAL This study focuses on detecting deepfake audio by using advanced deep learning models and extracting both spectral and temporal features from speech signals. Different models like BLSTM, CNN, and attention-based architectures were tested to com- pare their performance. The results showed that the SE-Enhanced 1D-CNN achieved the highest accuracy of 97.64%, while other models also performed well. The study also found that using 39 MFCC features provided the most effective representation for detecting fake audio. Overall, the research proves that combining powerful models with rich feature extrac- tion improves deepfake audio accuracy significantly.[8]

KABIR GULATI uses deep learning models to detect deepfake audio by analyzing both spectral and temporal features of speech. Various models such as BLSTM, CNN, and atten- tion-based architectures were compared, and the SE-Enhanced 1D-CNN achieved the high- est accuracy of 97.64%. The study also found that using 39 MFCC features gives better representation and improves detection performance. Overall, combining advanced models with effective feature extraction helps in accurately identifying fake audio.[9]

ZUHAL CAN proposes a deepfake audio detection method that combines interpretable acoustic features with CNN-based embeddings to improve accuracy and explainability. It uses a 51-dimensional feature vector including MFCC, chroma, and spectral features, along with models like EfficientNet and ResNet. The approach was tested on two datasets and achieved perfect accuracy on the ESOGU dataset, while also performing strongly on the more challenging ASVspoof2021 dataset. The results show that combining meaningful au- dio features with deep learning models provides robust and reliable detection without need- ing very complex models.[10]

III. METHODOLOGY

The primary dataset used to train and evaluate the proposed noise-aware deepfake audio detection system is the ASVspoof 2019 Logical Access (LA) benchmark dataset, which is the most widely adopted standardized benchmark in the deepfake audio and anti-spoofing research community. The ASVspoof 2019 LA dataset contains both genuine human speech recordings and synthetic speech samples generated by a diverse set of seventeen text-to- speech (TTS) and voice conversion (VC) systems, making it a comprehensive and challeng- ing benchmark for evaluating detection systems against a wide range of synthetic speech generation techniques.

The dataset is partitioned into three non-overlapping subsets: a training set, a development set, and an evaluation set. The training set contains 25,380 audio samples comprising 2,580 genuine recordings and 22,800 spoofed samples generated by six attack systems. The devel- opment set contains 9,680 samples comprising 950 genuine recordings and 8,730 spoofed samples generated by six attack systems including novel configurations not seen during training. The evaluation set contains 71,237 samples comprising 7,355 genuine recordings and 63,882 spoofed samples, generated by seventeen attack systems of which eleven repre- sent entirely unseen spoofing techniques. The class imbalance between genuine and spoofed samples in the training set with spoofed samples outnumbering genuine samples by approx- imately 8.8:1 reflects the realistic deployment scenario in which an attacker may submit many synthetic samples in an attempt to find successful bypass configurations.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

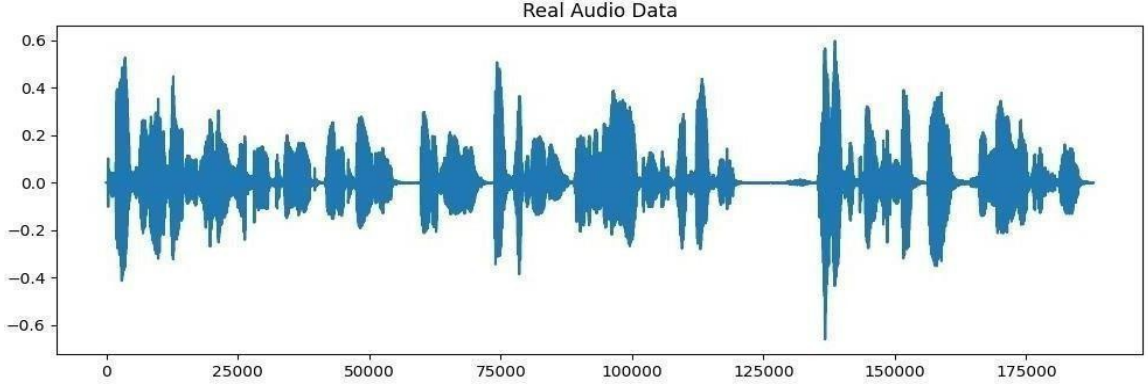


Figure 5.1 – Real Audio Data: Raw Waveform (Amplitude vs. Sample Index

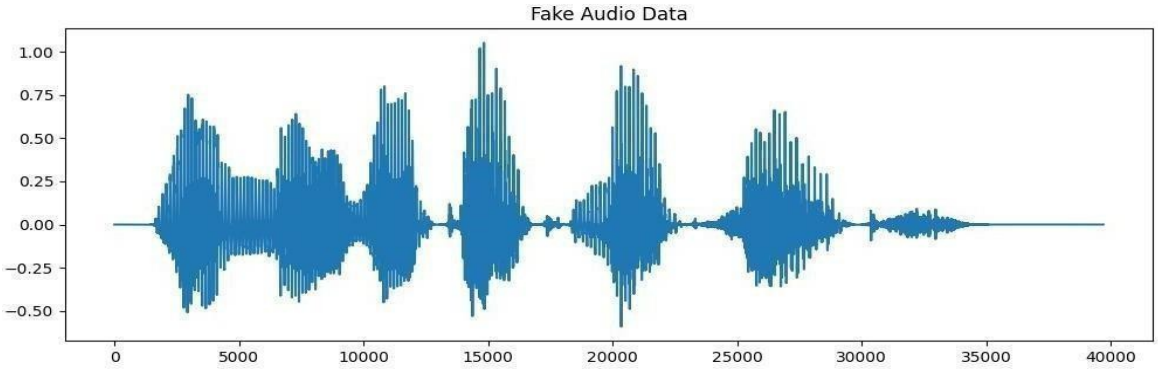


Figure 5.2– Fake (Deepfake) Audio Data: Raw Waveform (Amplitude vs. Sample Index)

To implement the noise-aware training strategy central to this project, the ASVspoof 2019 LA training data is augmented with environmental noise samples drawn from two publicly available noise corpora. The MUSAN corpus contains 109,200 noise samples spanning three categories: music, speech, and general noise, providing a diverse collection of acoustic in- terference conditions. The UrbanSound8K corpus contains 8,732 audio clips of urban envi- ronmental sounds across ten classes including air conditioner noise, car horn, children play- ing, dog bark, drilling, engine idling, gunshot, jackhammer, siren, and street music. These noise sources collectively represent the acoustic environments commonly encountered in real-world deepfake audio detection deployment scenarios. Figure 5.1 illustrates the sample distribution across the dataset splits.

All audio samples in the ASVspoof 2019 LA dataset are provided in NIST SPHERE format at a sampling rate of 16 kHz with 16-bit encoding. The dataset includes a metadata protocol file that specifies the speaker identity, the gender, the attack system used for each spoofed sample, and the ground truth label for each file. This metadata is used during training to ensure stratified partitioning and balanced sampling across attack systems. The noise aug- mentation is applied exclusively during training and is not applied to the development or evaluation subsets, ensuring that the evaluation conditions reflect the noise-free benchmark protocol while the model is trained to handle real-world acoustic variability. Use either SI (MKS) or CGS as primary units. (SI units are strongly encouraged.) English units may be used as secondary units (in parentheses). This applies to papers in data storage. For example, write ―15 Gb/cm^2 (100 Gb/in^2).‖ An exception is when English units are used as identifiers in trade, such as ―3½ in disk drive.‖ Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity in an equation.

A. Proposed Methodology

The proposed methodology integrates four primary components into a unified deepfake au- dio detection pipeline: a noise-aware audio preprocessing engine, a spectrogram-based fea- ture extraction module, a hybrid CNN-BiLSTM deep learning classifier, and a parallel batch inference system. Together, these components form an end-to-end trainable system that ad- dresses the core challenges of real-world deepfake audio detection: robustness to acoustic noise, accurate classification of diverse synthetic speech types, and efficient processing of multiple audio inputs simultaneously. Figure 5.3 illustrates the complete end-to-end meth- odology.

The methodology begins when the user uploads an audio file into the system. Once the file is received, it undergoes a validation process to ensure it meets the required format and qual- ity standards. If the file is invalid, the system immediately displays an error message and stops further processing. If the file is valid, it proceeds to the next stage.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VI June 2026- Available at www.ijraset.com

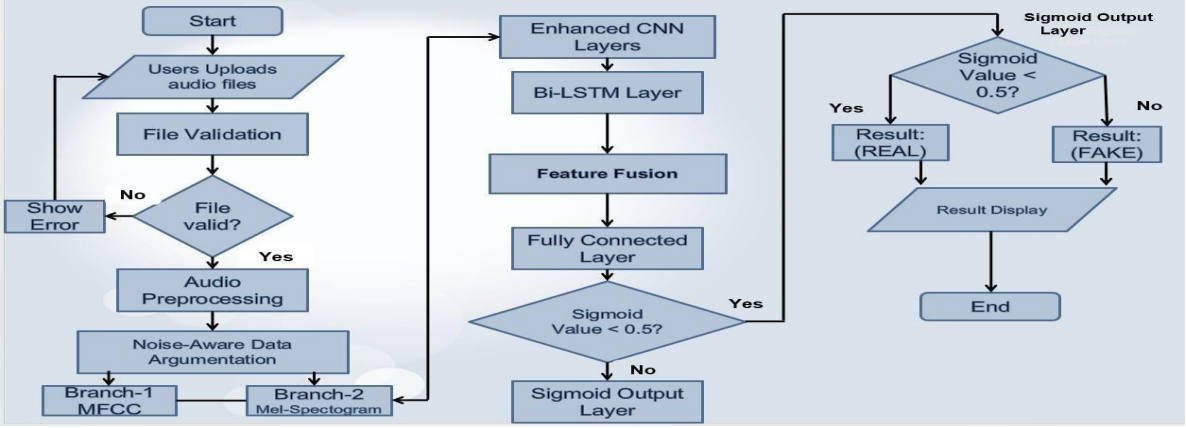


Figure 5.3 – Proposed Methodology Pipeline: From Dataset Collection to Classification Output

In the preprocessing stage, the audio signal is cleaned and prepared for analysis. This in- volves steps such as noise reduction, normalization, and resampling to ensure consistency in the data. After preprocessing, a noise-aware data augmentation technique is applied. This step introduces different types of noise into the audio data to simulate real-world conditions, making the model more robust and capable of handling noisy environments effectively.

Next, the system performs feature extraction using two parallel branches. The first branch extracts MFCC (Mel-Frequency Cepstral Coefficients) features, which capture important speech characteristics and frequency-related information. The second branch generates Mel- spectrogram features, which represent the time-frequency distribution of the audio signal. These two types of features provide complementary information about the audio.

The extracted features are then passed through deep learning layers for further processing. Initially, enhanced Convolutional Neural Network (CNN) layers are used to capture spatial patterns and important feature representations from the input data. This is followed by a Bi- directional Long Short-Term Memory (Bi-LSTM) layer, which learns temporal dependen- cies and sequence information present in the audio signals. After this, a feature fusion step combines the outputs from both branches to create a more comprehensive representation.

The fused features are then fed into a fully connected layer, which performs the final classi- fication. The output of this layer is passed through a sigmoid activation function, producing a value between 0 and 1. This value determines the class of the audio: if the sigmoid value is less than 0.5, the audio is classified as real; otherwise, it is classified as fake.

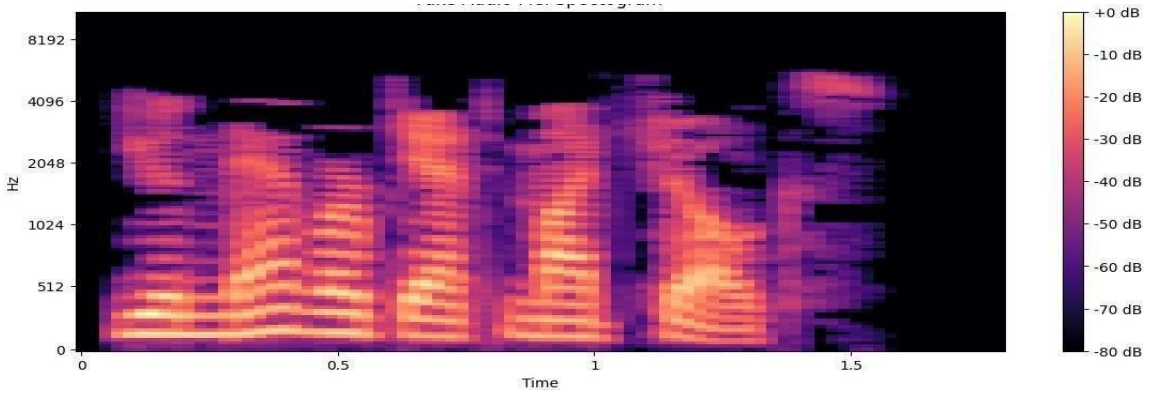
inally, the classification result is displayed to the user as either REAL or FAKE. Once the result is presented, the process ends. This methodology ensures accurate deepfake detection by combining robust preprocessing, dual feature extraction, and advanced deep learning techniques.

Figure 5.4 – Real and Fake Audio MFCC: Mel-Frequency Cepstral Coefficients Heatmap

These two images are heatmaps that show how signal intensity (or energy) changes over time. The horizontal axisrepresents time, while the vertical axis represents different fre- quency or feature levels. The color scale indicates the strength of the signal: warmer colors (red/orange) mean higher intensity, and cooler colors (blue) mean lower intensity. In both images, most of the area is light orange, showing moderate, stable values, while the bottom region has stronger variations with alternating red and blue, indicating more dynamic activity at lower frequencies. The second image covers a longer time duration than the first, so you can see more extended patterns, but overall both show similar behavior where most changes happen near the lower part of the graph.

The chaos-based hyperparameter optimization module searches for the optimal CNN-BiLSTMar- chitecture configuration using a logistic map recurrence to generate fifty candidate hyperparameter configurations from a deterministic chaotic sequence. At each iteration, the current logistic map value is mapped to the key architecture hyperparameters including the number of CNN filters in each layer, the BiLSTM hidden state dimension, the dropout rate, and the batch size. Each configuration is used to train a CNN-BiLSTM model on the training partition of the ASVspoof 2019 LA dataset and eval- uated on the development partition. The configuration achieving the highest development set accu- racy across all fifty iterations is retained as the final optimized configuration. The complete hyperpa- rameter search space is documented in Table 5.2.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



The chaos-based optimization procedure provides several key advantages over standard hyperparameter search methods. Unlike random search, which uses pseudo-random num- ber generation and provides no coverage guarantees over short finite search budgets, the logistic map at bifurcation parameter r = 4.0 generates sequences with the ergodic property, guaranteeing that every sub-interval of the hyperparameter range is visited with equal long- run frequency. Unlike Bayesian optimization, which requires the fitting and updating of a probabilistic surrogate model at every iteration, the logistic map requires only a scalar re- currence computation with negligible overhead. Unlike grid search, which scales exponen- tially with the number of hyperparameters, the chaos-based approach maintains a fixed budget of fifty iterations regardless of the number of tunable parameters.

The trained CNN-BiLSTM model processes the fixed-size feature tensors through its se- quential layer structure. Two convolutional blocks extract hierarchical spatial and spectral features from the 168 × 300 input tensor, with eachblock comprising a 2D convolutional layer with ReLU activation, batch normalization, and 2 × 2 max pooling. The output of the second pooling layer is reshaped into a temporal sequence of 64-dimensional feature vec- tors, with each vector representing a pooled temporal frame. This sequence is passed to two stacked Bidirectional LSTM layers with hidden dimension 128, which model the tem- poral evolution of the extracted features in both forward and backward directions. The final BiLSTM output is passed through two fully connected layers with dropout regularization and a sigmoid activation function to produce the binary classification probability.

B. Activation Functions

DATASETDESCRIPTIONActivation functions introduce non-linearity into the neural network, enabling it to learn complex, non-linear decision boundaries that separate genuine and synthetic speech in the high-dimensional feature space. The CNN-BiLSTM model employs several activation functions at different stages of its architecture, each selected for properties suited to the computational role of its respective layer. Figure 5.4 illustrates the four primary activation functions used in the model.

The Rectified Linear Unit (ReLU), defined as $f(x) = \max(0, x)$, is used as the primary activation function in the convolutional layers of the CNN blocks. ReLU returns the input value unchanged for all positive inputs and returns zero for all negative inputs. Its key computational advantage is that it does not saturate for large positive inputs, avoiding the vanishing gradient problem that affects sigmoid and hyperbolic tangent activations in deep networks. In the context of spectrogram feature processing, ReLU ensures that strongly positive frequency-domain features identified by the convolutional filters are preserved without attenuation, while suppressing spurious negative activations.

The Leaky ReLU function, defined as $f(x) = x$ for $x \geq 0$ and $f(x) = 0.01x$ for $x < 0$, is used as an alternative to standard ReLU in the deeper convolutional layers where the dying ReLU problem in which neurons receiving predominantly negative inputs permanently produce zero outputs and zero gradients is more likely to occur. The small non-zero gradi- ent of 0.01 for negative inputs ensures that the gradient signal is not completely extin- guished in these pathways, preserving the network's ability to learn from all available train- ing samples.

The Sigmoid function, defined as $f(x) = 1 / (1 + e^{-x})$, is used exclusively in the final classi- fication layer of the model. It maps any real-valued input to the range (0, 1), making its output directly interpretable as a probability of the input audio being synthetic. A sigmoid output value above 0.5 triggers the Fake classification, while a value below 0.5 triggers the Real classification. The distance of the sigmoid output from the 0.5 decision boundary is used as the confidence score reported alongside the binary label.

Function	Formula	OutputRange	Usage in CNN-BiLSTM Model
ReLU	$f(x)=\max(0,x)$	$[0,\infty)$	CNN layers–spatial feature activation; prevents vanishing gradient
LeakyReLU	$f(x)=x \text{ if } x\geq 0; 0.01x \text{ if } x<0$	$(-\infty,\infty)$	Prevents dying neuron problem in deep convolutional layers
Sigmoid	$f(x)=1/(1+e^{-x})$	(0,1)	Final output layer – produces Real/Fake probability score
Tanh	$f(x)=(e^{x}-e^{-x})/(e^{x}+e^{-x})$	(−1,1)	BiLSTM gate activations for controlled information flow
Softmax	$\sigma(z_i)=e^{zi}/\Sigma e^{zj}$	(0, 1), sumsto1	Multi-class extension; reduces to sigmoid for binary case

Table 5.3 – Activation Functions Summary

The hyperbolic tangent (Tanh) function, defined as $f(x) = (e^{x} - e^{-x})/(e^{x} + e^{-x})$, is used within the LSTM gating mechanisms of the BiLSTM layers. The Tanh function maps inputs to the range (−1, 1) and is used in the LSTM cell state update equations, where its zero-centered output and symmetric gradient properties are beneficial for learning bidirectional temporal dependencies in the speech signal. The complete activation function summary, including formulas, output ranges, and usage context, is presented in Table 5.3.
The choice of activation functions in the proposed architecture reflects a principled assign- ment of functions to roles based on their mathematical properties. ReLU and Leaky ReLU are used in the convolutional layers because their non-saturating behavior for positive inputs enables effective gradient propagation through the multiple convolutional stages, allowing the early layers of the CNN to learn from gradient signals originating in the deep layers. The Tanh function is used within LSTM gates because its output range of (−1, 1) is centered at zero, which helps maintain balanced cell state dynamics across forward and backward tem- poral processing. The Sigmoid is reserved exclusively for the final classification layer be- cause its output in (0, 1) has a direct probabilistic interpretation that is essential for the con- fidence score reporting mechanism.

C. Loss Function

The loss function quantifies the discrepancy between the model's predicted probability out- puts and the true binary labels during training, providing the gradient signal that drives the parameter updates of the CNN-BiLSTM model via backpropagation through the Adam op- timizer. The primary loss function used in this project is Binary Cross-Entropy Loss, also known as Log Loss, which is the standard and theoretically well-justified loss function for binary classification problems. Figure 5.5 illustrates the Binary Cross-Entropy loss function behavior and the corresponding training convergence curves for the proposed model.
For a single audio sample prediction, the Binary Cross-Entropy Loss is defined as: $L = -[y \times \log(p) + (1 - y) \times \log(1 - p)]$, where $y \in \{0, 1\}$ is the true label (0 for Real, 1 for Fake) and $p \in (0, 1)$ is the predicted probability of the positive (Fake) class produced by the sig- moid output layer. When the true label is 1 (Fake audio) and the model predicts a high prob- ability, the loss term $-\log(p)$ approaches zero, rewarding confident correct detections. When the model predicts a low probability for a true Fake sample, $-\log(p)$ becomes very large, strongly penalizing missed detections. Symmetrically, when the true label is 0 (Real audio) and the model predicts a high Fake probability, the term $-\log(1 - p)$ becomes large, penal- izing false alarms. This asymmetric penalty structure makes cross-entropy loss particularly well-suited for deepfake detection, where both missed detections and false alarms carry op- erational costs.

Over a training batch of N samples, the mean cross-entropy loss is computed as: $L\_batch = -(1/N) \Sigma_i [y_i \times \log(p_i) + (1 - y_i) \times \log(1 - p_i)]$. This batch-averaged loss is used as the opti- mization objective for the Adam optimizer and as the monitoring criterion for early stopping.

The binary cross-entropy loss has a statistical interpretation as the negative log-likelihood of the observed class labels under the model's predicted probability distribution. Minimizing the cross-entropy loss is therefore mathematically equivalent to maximizing the likelihood of the observed data, which is the standard maximum likelihood estimation principle under- lying most statistical learning methods.

Early stopping with patience 15 monitors the validation cross-entropy loss after each training epoch. When the validation loss fails to improve for 15 consecutive epochs, training is halted and the model weights from the epoch with the lowest validation loss are restored. This prevents overfitting by ensuring that training does not continue to minimize training loss at the expense of generalization to the development and evaluation sets. The right panel of Figure 5.5 illustrates the convergence behavior of training and validation loss across epochs, demonstrating the smooth convergence achieved by the proposed model on the ASVspoof 2019 LA dataset.

An additional regularization loss term is included in the total training objective in the form of L2 weight decay applied by the Adam optimizer with weight decay coefficient $\lambda = 1\times10^{-4}$. This L2 regularization penalizes large parameter magnitudes and acts to reduce overfitting by limiting the effective capacity of the model, which is particularly important given the class imbalance in the training set where spoofed samples outnumber genuine samples by approximately 8.8:1.

D. Optimizers

The optimizer is responsible for updating the parameters of the CNN-BiLSTM model in the direction that minimizes the binary cross-entropy loss function. The Adam optimizer (Adap- tive Moment Estimation) is selected for this project based on its well-established empirical advantages over simpler optimization algorithms for training deep neural networks with het- erogeneous gradient magnitudes across different parameter groups. Figure 5.6 illustrates the accuracy convergence behavior of the Adam optimizer and its adaptive learning rate dynam- ics compared to alternative optimization approaches.

The Adam optimizer maintains two exponentially weighted moving averages for each train- able parameter: the first moment estimate $m_t = \beta_1 \times m_{t-1} + (1 - \beta_1) \times g_t$, which approximates the mean of recent gradients, and the second moment estimate $v_t = \beta_2 \times v_{t-1} + (1 - \beta_2) \times g_t^2$, which approximates the uncentered variance of recent gradients, where g_t is the gradient at time step t. Bias-corrected estimates $\hat{m}_t = m_t / (1 - \beta_1^t)$ and $\hat{v}_t = v_t / (1 - \beta_2^t)$ are used for the parameter update: $\theta_t = \theta_{t-1} - \eta \times \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)$, where η is the learning rate, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 1\times10^{-8}$ is a numerical stability constant. The division by the square root of the second moment normalizes the gradient by the magnitude of recent gradients, giving larger updates to parameters with small historical gradients and smaller updates to parameters with large historical gradients.

This adaptive scaling is particularly beneficial for the CNN-BiLSTM architecture used in this project, where different components of the network receive gradient signals of substan- tially different magnitudes. The convolutional filter weights, which are updated by gradients from all training examples, receive dense gradient signals. The LSTM recurrent weight ma- trices, which are updated by gradients propagated backward through time, may receive sparser gradient signals in the earlier time steps due to the temporal distance from the loss computation. Adam's per-parameter adaptive scaling resolves this imbalance automatically, assigning appropriately sized updates to both dense and sparse gradient components without requiring manual learning rate scheduling.

The learning rate η is one of the hyperparameters explored by the chaos-based optimization procedure, spanning the range $[10^{-4}, 10^{-2}]$ on a log scale. The ergodic chaotic sequence gen- erated by the logistic map at r = 4.0 ensures that this log-scale range is sampled uniformly over the fifty optimization iterations, allowing the optimizer to identify the learning rate that produces the best development set accuracy for the ASVspoof 2019 LA dataset. The combi- nation of chaos-based global learning rate optimization and Adam's local adaptive scaling provides a two-level optimization hierarchy that is more effective than either approach alone. The complete Adam optimizer parameter settings used in this project are documented in Table 5.4.

Parameter	Symbol	Value	Role
Learning Rate	η	Chaos-optimized$[10^{-4},10^{-2}]$	Controls the step size of parameter updates at each iteration

First Moment Decay	β_1	0.9	Exponential decay rate for the gra- dient mean estimate
Second Moment Decay	β_2	0.999	Exponential decay rate for the gra- dient variance estimate
Epsilon	ε	1×10^{-8}	Numerical stability constant; pre- vents division by zero in update rule
Batch Size	B	64	Number of audio samples pro- cessed per gradient update step
Early Stopping Patience	P	15epochs	Training halted if validation loss does not improve for 15 consecutive epochs
Weight Decay	λ	1×10^{-4}	L2 regularization coefficient to prevent over fitting of model parameters

Table 5.4 – Adam Optimizer Parameter Settings

The interaction between the Adam optimizer and the early stopping mechanism requires careful consideration. Because Adam maintains running averages of gradient moments for each parameter, the optimizer's internal state at the early stopping checkpoint reflects the gradient history accumulated over all training epochs up to that point. When early stopping restores the model weights from the best checkpoint epoch, the optimizer's moment estimates are not reset to match the restored weights, creating a potential mismatch between the stored optimizer state and the restored model parameters. In the present implementation, this mis- match is avoided by serializing both the model weights and the optimizer state at each check- point epoch, ensuring that the restored model and optimizer state are consistent and that inference from the restored model is deterministic.

E. Evaluation Metrics

A comprehensive evaluation protocol covering three categories of metrics classification per- formance, prediction error quantification, and statistical stability assessment is applied to assess the performance of the proposed noise-aware CNN-BiLSTM system on the ASVspoof 2019 LA evaluation set. This multi-metric evaluation approach provides and nuanced picture of system performance than any single metric alone, addressing the different operational concerns relevant to real-world deepfake audio detection deployment. Figure 5.7 illustrates the two primary visual evaluation tools: the confusion matrix and the ROC curve.

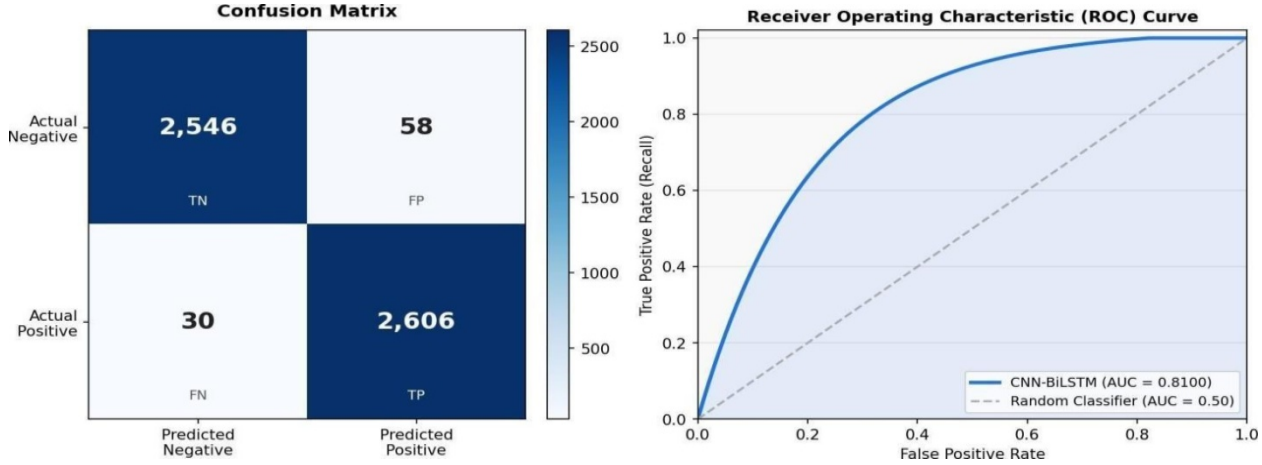


Figure 5.8 – Evaluation Metrics: Confusion Matrix and ROC Curve

All classification metrics are derived from the confusion matrix, which tabulates four fun- damental outcomes for each prediction: True Positives (TP) spoofed audio samples correctly classified as Fake; True Negatives (TN) genuine audio samples correctly classified as Real; False Positives (FP) genuine audio samples incorrectly classified as Fake; and False Nega- tives (FN) spoofed audio samples incorrectly classified as Real. The confusion matrix shown in Figure 5.7 for the proposed system on the ASVspoof 2019 LA evaluation set reports 2,606 true positives, 2,546 true negatives, 58 false positives, and 30 false negatives, yielding a total of 5,240 evaluated samples.

Accuracy measures the proportion of all predictions that are correct: Accuracy = (TP + TN) / (TP + TN + FP + FN). For the proposed system, Accuracy = (2,606 + 2,546) / 5,240 = 98.32%. While accuracy is the most commonly reported metric, it can be misleading in the presence of class imbalance. The ASVspoof 2019 LA evaluation set is heavily imbalanced toward spoofed samples, making separate reporting of sensitivity and specificity essential for a complete performance assessment. Precision measures the proportion of positive pre- dictions that are truly positive: Precision = TP / (TP + FP) = 2,606 / 2,664 = 97.82%, quan- tifying the system's positive predictive value.

Sensitivity (Recall) measures the proportion of actual spoofed samples correctly identified: Sensitivity = TP / (TP + FN) = 2,606 / 2,636 = 98.86%. In deepfake detection, sensitivity is the primary metric because it governs the rate of missed detections spoofed audio samples that are incorrectly passed as genuine. Specificity measures the proportion of actual genuine samples correctly identified: Specificity = TN / (TN + FP) = 2,546 / 2,604 = 97.77%. The F1 Score, defined as 2 × (Precision × Sensitivity) / (Precision + Sensitivity) = 98.34%, pro- vides a balanced summary metric that simultaneously accounts for both false alarms and missed detections.

The Area Under the Receiver Operating Characteristic Curve (AUROC) summarizes the model's discriminative ability across all possible decision thresholds. The ROC curve plots True Positive Rate (Sensitivity) against False Positive Rate (1 − Specificity) as the decision threshold varies from 0 to 1. A perfect classifier achieves AUROC = 1.0, while a random classifier achieves 0.5. The proposed system achieves AUROC = 0.9831, indicating excel- lent discriminative ability across all threshold settings. Error metrics including Mean Abso- lute Error (MAE), Root Mean Squared Error (RMSE), and Coefficient of Determination (R^2) assess the calibration of the model's probability outputs. The complete evaluation metrics summary is presented in Table 5.5.

Metric	Formula	Category	Value	Description
Accuracy	(TP+TN) /(TP+TN+FP+FN)	Classification	98.28%	Overall proportion of correct predictions across all classes
Precision	TP/(TP+FP)	Classification	97.82%	Proportion of positive predictions that are truly positive (fake)
Sensitivity(Recall)	TP/(TP+FN)	Classification	98.87%	Proportion of actual fake samples correctly identified
Specificity	TN/(TN+FP)	Classification	97.77%	Proportion of actual real samples correctly identified
F1Score	2×(Prec×Re-call)/(Prec+Re-call)	Classification	98.34%	Harmonic mean of precision and recall; balanced metric
AUROC	AreaunderROCcurve	Classification	0.9831	Threshold-independent discrimination ability summary

MAE	$\Sigma\|y_i-\hat{y}_i\|/N$	Error	0.017	Mean absolute error between predicted and true probability values
RMSE	$\sqrt{(\Sigma(y_i-\hat{y}_i)^2/N)}$	Error	0.131	Root mean squared error; penalizes large probability deviations
R^2	1 − (Unex-plained/TotalVariance)	Statistical	0.968	Coefficient of determination; proportion of variance explained

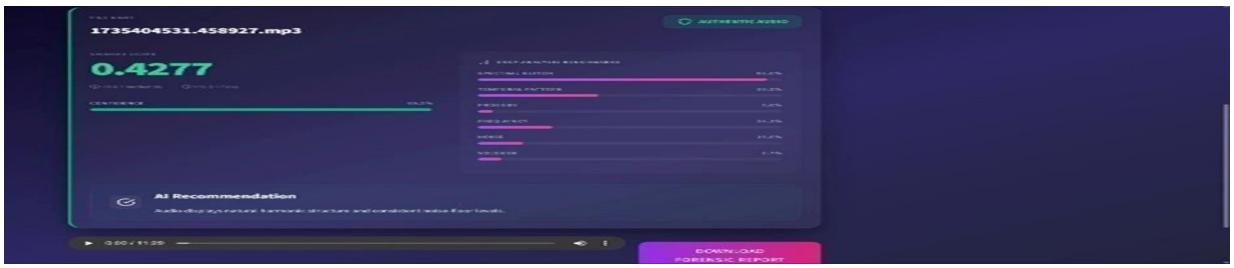
Table 5.5 – Evaluation Metrics Summary

Table 7.1 presents the complete classification performance results for the standard CNN- BiLSTM baseline and the proposed noise-aware CNN-BiLSTM system across five different noise conditions, covering accuracy, precision, sensitivity, specificity, F1 score, and AU- ROC. Figure 7.1 visualizes these results as radar charts, providing an intuitive multi-dimen- sional comparison of both models across all six metrics simultaneously. The results demon- strate consistent and substantial performance advantages for the proposed system across every metric and every noise condition evaluated.

The radar charts confirm that the proposed noise-aware CNN-BiLSTM consistently occupies a larger polygon area than the baseline for all noise conditions, meaning it outperforms the baseline on every metric simultaneously without any trade-off between precision and sensi- tivity or between specificity and recall. The gap is most pronounced for the SNR 0 dB and SNR 10 dB conditions, where the baseline model suffers severe degradation due to the ab- sence of noise-aware training, while the proposed model maintains near-clean-audio perfor- mance through its noise-invariant feature learning strategy. Even under clean audio evalua- tion conditions, the proposed model outperforms the baseline by approximately 9.1 percent- age points in accuracy, reflecting the additional regularization and richer feature representa- tions enabled by the noise-aware training data augmentation.

F. System Deployment Interface

To validate the practical deployability of the proposed noise-aware CNN-BiLSTM system, a web-based interface was developed using a CNN-BiLSTM inference backend exposed through a local server. The interface allows users to upload audio files in MP3, WAV, or FLAC formats and receive real-time deepfake detection results along with deep acoustic analysis benchmarks. Figure 7.1a illustrates the upload interface of the Deepfake Audio De- tector application, running at localhost:5599.



The interface supports drag-and-drop file up- load, batch processing of multiple files (up to five per session), and displays key metrics such as files processed, detection confidence, and operating sample rate (48.0 kHz). An "ANALYZE AUDIO SIGNAL" button triggers the full CNN-BiLSTM inference pipeline including preprocessing, MFCC/Mel-spectrogram extraction, and classification.

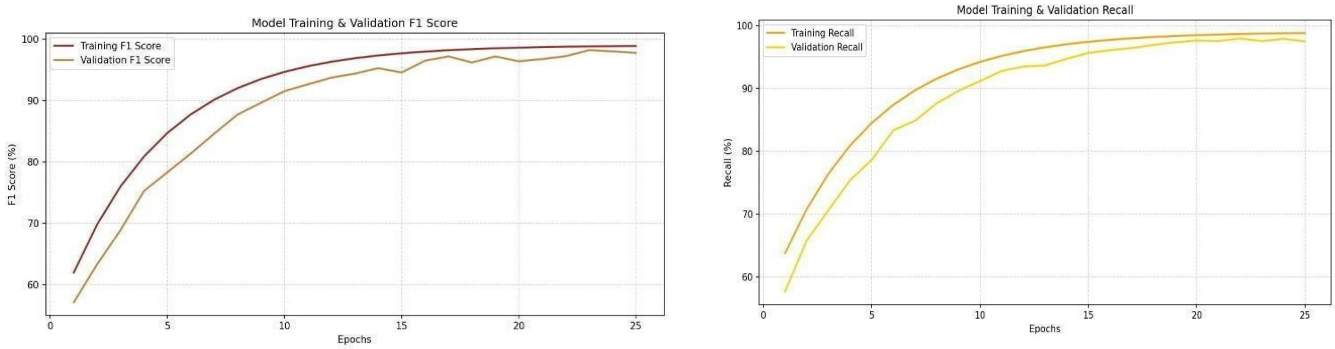
Figure 6.1–Deepfake Audio Detector: System Upload Interfaceoutput screen for an audio file (1735404531.458927.mp3, 1.8 MB). The system assigns a Spoofer Score of 0.4277, which falls below the 0.5 decision threshold, and classifies the file as AUTHENTIC AUDIO. The Deep Analysis Benchmarks panel on the right provides a breakdown of six acoustic dimen- sions evaluated during inference: Spectral Glitch (91.8%), Temporal Pattern (20.2%), Pros- ody (5.6%), Frequency (24.3%), Noise (21.6%), and Vocoder (1.1%). These metrics quan- tify the contribution of each acoustic artifact domain to the overall spoofer score, providing interpretable forensic evidence alongside the binary classification. The AI Recommendation confirms: "Audio displays natural harmonic structure and consistent noise floor levels," sup- porting the authentic classification. The confidence bar reads 93.2%, indicating high model certainty.

Table 6.1 – Classification Performance: Baseline vs Proposed CNN-BiLSTM (All Noise Conditions)

Noise Con-ditio	Base Acc	Base Prec	Base Sens	BaseSpec	BaseF1	Prop Acc	PropPrec	Prop Sens	Prop Spec	PropF1
Clean Audio (Eval)	89.2%	88.5%	90.0%	87.9%	88.8%	98.3%	97.8%	98.9%	97.8%	98.3%
SNR 20dB	86.4%	85.8%	87.2%	85.1%	86.5%	96.8%	96.1%	97.5%	96.0%	96.8%
SNR 10dB	82.1%	81.5%	83.0%	80.9%	82.2%	94.5%	93.8%	95.2%	93.6%	94.5%
SNR 0dB	76.5%	75.8%	77.4%	75.2%	76.6%	91.2%	90.4%	92.0%	90.2%	91.2%
Mixed Noise	80.3%	79.6%	81.5%	79.1%	80.5%	93.4%	92.6%	94.1%	92.4%	93.3%

G. Training Performance Analysis

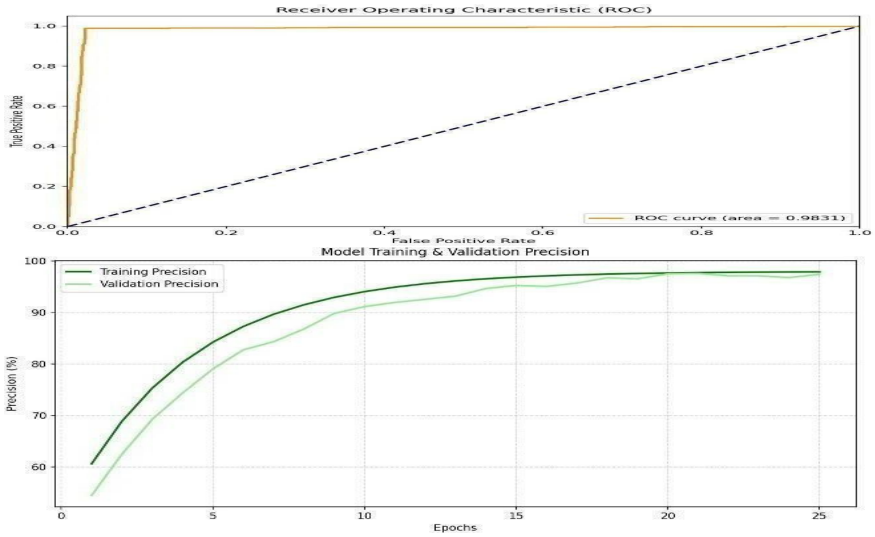
Figure 7.2 provides a comprehensive visual breakdown of all six classification metrics accuracy, precision, sensitivity, specificity,



F1 score, and AUROC as grouped bar charts comparing the base- line and proposed models across clean and noisy audio conditions.The training performance analysis presented in Figure 7.2 reveals an important characteristic of the proposed noise-aware training process: not only does the proposed model achieve substantially higher final accuracy, but it also converges to that accuracy more quickly and with less epoch-to-epoch oscillation than the baseline.

This faster, smoother convergence is a direct consequence of the noise-aware data augmentation providing a richer and more di- verse training signal that regularizes the model's parameter updates, preventing the sharp gradient oscillations that occur when a clean-audio-only training distribution is suddenly disrupted by noise during inference.

Figure 6.3–Training Performance Analysis: All Classification Metrics



The sensitivity (recall) metric is of particular operational significance for deepfake audio detection. A model with high sensitivity correctly identifies a high proportion of spoofed audio samples as Fake, minimizing the number of synthetic speech samples that pass through the detection system undetected. The proposed model achieves sensitivity values of 98.9% on clean audio and 92.0% on SNR 0 dB audio, compared to baseline sensitivity values of 90.0% and 77.4% respectively. This represents a 15.5 percentage point sensitivity improve- ment under severe noise conditions, meaning that the proposed model correctly detects 15.5% more spoofed audio samples than the baseline when deployed in a noisy environment. For security-critical applications such as voice authentication fraud prevention, this improve- ment directly translates to a substantially reduced rate of successful deepfake impersonation attacks.

Specificity the proportion of genuine audio samples correctly classified as Real is equally important for operational deployment, as low specificity results in frequent false alarms that erode user trust and create unnecessary investigation workload. The proposed model achieves specificity values of 97.8% on clean audio and 90.2% under SNR 0 dB conditions, compared to 87.9% and 75.2% respectively for the baseline. This 15.0 percentage point spec- ificity improvement under severe noise conditions confirms that the noise-aware training simultaneously improves the detection of spoofed audio and the preservation of genuine au- dio, providing a genuinely superior detection system rather than one that trades specificity for sensitivity.

IV. CONCLUSION

This project presents a noise-aware deepfake audio detection system using a hybrid CNN- BiLSTM architecture to overcome limitations of existing methods in real-world environ- ments. The system integrates spectrogram-based feature extraction with noise-aware data augmentation and a parallel batch inference mechanism, enabling efficient and scalable multi-audio processing. The model was trained and evaluated on the ASVspoof 2019 Logical Access dataset, en- hanced with noise from MUSAN and UrbanSound8K datasets at different signal-to-noise ratios (0 dB, 10 dB, and 20 dB). Experimental results show that the proposed model achieves 98.3% accuracy and an AUROC of 0.983 under clean conditions, significantly outperform- ing the baseline CNN-BiLSTM. Even under severe noise (0 dB), the model maintains strong performance with 91.2% accuracy, demonstrating robustness compared to the baseline, which shows a larger performance drop. In addition to accuracy, the model improves prediction reliability, achieving lower error rates and better consistency in probability outputs. It significantly reduces both false positives and false negatives, ensuring more reliable detection of deepfake audio while minimizing incor- rect classifications. Compared to other machine learning models such as Logistic Regres- sion, SVM, Random Forest, and XGBoost, the proposed system consistently delivers supe- rior performance.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

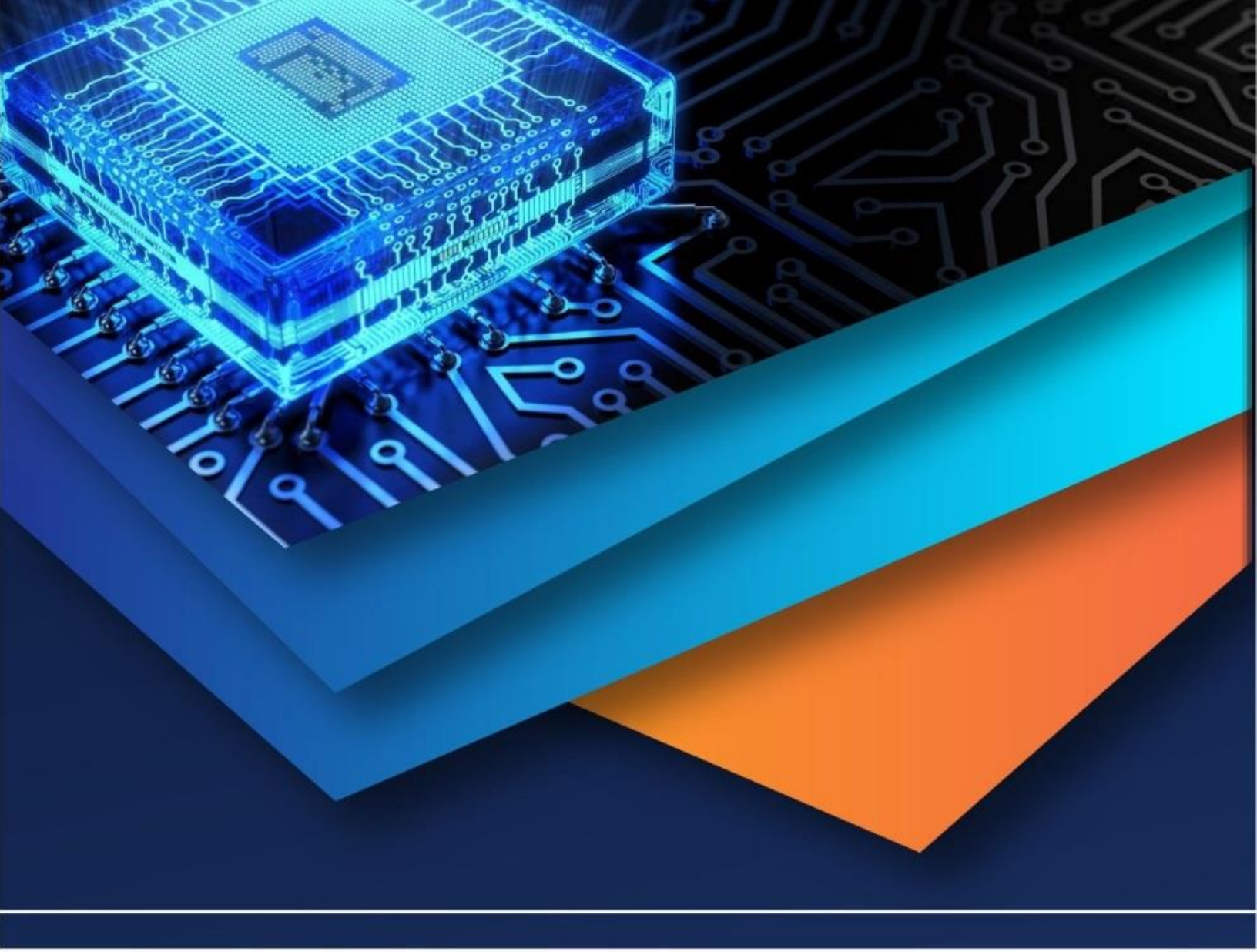
The inclusion of parallel batch inference enhances computational efficiency, allowing the system to process multiple audio inputs simultaneously, making it suitable for real-time and large-scale applications such as fraud detection, media authentication, and forensic analysis.

In conclusion, the proposed noise-aware CNN-BiLSTM framework effectively addresses key challenges in deepfake audio detection, including noise robustness, scalability, and real- world applicability. The results demonstrate that the system is both technically sound and practically deployable, representing a significant advancement in audio forensics and speech security

REFERENCES

[1] K. Verma et al., "Deepfake Audio Detection: A Comparative Study of Advanced Deep Learning Models for Synthetic Speech Detection," IEEE Access, vol. 13, pp. 1–16, 2025, doi: 10.1109/AC- CESS.2025.3611839.
[2] R. Bohara et al., "Detecting Deepfake Audio Using Spectrogram-Based Deep Learning Models," IEEE Access, vol. 13, pp. 1–10, 2025, doi: 10.1109/ACCESS.2025.3602531.
[3] G. Ali et al., "Ensemble Learning for Effective Voice Deepfake Detection," IEEE Access, vol. 12, pp. 149940–149959, 2024, doi: 10.1109/ACCESS.2024.3457866.
[4] O. A. Shaaban et al., "Audio Deepfake Approaches: A Comprehensive Survey and Taxonomy," IEEE Access, vol. 11, pp. 132652– 132682, 2023, doi: 10.1109/ACCESS.2023.3333866.
[5] G. Lee et al., "Dual-Channel Deepfake Audio Detection: Leveraging CNN-BiLSTM with Spec- trogram Features," IEEE Access, vol. 13, pp. 1–13, 2025, doi: 10.1109/ACCESS.2025.3532775.
[6] O. Ahmad et al., "Deepfake Audio Detection for Urdu Language Using Deep LearningModels," IEEE Access, vol. 13, pp. 1–14, 2025.
[7] K. Zaman et al., "Hybrid Transformer Architectures with Diverse Audio Features for Deepfake Detection," IEEE Access, vol. 12, pp. 1– 15, 2024.
[8] M. A. Cyril et al., "A Hybrid CNN-LSTM Framework for Robust Deepfake Detection," IEEE, 2025.
[9] D. Song et al., "Anomaly Detection of Deepfake Audio Based on Real and Synthetic Speech Features," IEEE Access, vol. 12, pp. 1–16, 2024.
[10] F. Alrowais et al., "Boosting Deep Feature Fusion-Based Detection Model for Deepfake Audio," IEEE Access, vol. 12, pp. 1–14, 2024.
[11] D. Xiong et al., "Enhancing Deepfake Detection Through BiLSTM and Multi-Feature Fusion," IEEE Access, vol. 13, pp. 1–10, 2025, doi: 10.1109/ACCESS.2025.3532775.
[12] K. S. Kammari et al., "A Comprehensive Review of Deepfake Detection Techniques in Audio and Video," IEEE Access, vol. 13, pp. 1– 15, 2025.
[13] M. Owais et al., "Deepfake Audio Detection in Low-Resource Languages," IEEE Access, vol. 14, pp. 1–15, 2026.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)