Isabela State University Linker:
Journal of Engineering, Computing, and Technology
Volume 2, Issue 1
ISSN (Print): 3082-3676 ISSN (Online): 3082-3684
Publisher: Isabela State University, Desktop Publishing, Echague, Isabela

Deepfake Speech Detection: Identifying AI-Generated and Real Human Voices Using Hybrid Convolutional Neural Network and Long Short-Term Memory Model

Marc P. Laureta[1], **John Maynardk M. Atienza**[2], **John Lemuel B. Tapel**[3]
College of Informatics and Computing Studies, New Era University, Quezon City, 1107, Philippines [1,2,3]
mplaureta@neu.edu.ph; johnmaynardk.atienza@neu.edu.ph; johnlemuel.tapel@neu.edu.ph

RESEARCH ARTICLE INFORMATION

Received: April 16, 2025
Reviewed: May 23, 2025
Accepted: June 16, 2025
Published: June 30, 2025

Copyright © 2025 by the Author(s). This open-access article is distributed under the Creative Commons Attribution 4.0 International License.

ABSTRACT

This study explored deepfake audio detection using English and Tagalog datasets to enhance multilingual speech classification. The rise of synthetic media, particularly deepfake audio, raises concerns about misinformation, security, and authenticity. To address this, the researchers developed a web-based detection system using a hybrid Convolutional Neural Network and Long Short-Term Memory Model (CNN-LSTM) model, which captured spatial and temporal features for accurate classification. The approach leveraged Mel spectrograms, convolutional layers for spatial patterns, and LSTM networks for temporal dependencies. Trained on an augmented dataset of over 176,000 samples and fine-tuned using TensorFlow, the model achieved 98.65% accuracy, with a precision of 98.60% and a recall of 98.76%. The system employed class weighting to address imbalance and used mixed-precision training for efficiency. Its architecture included Conv2D layers with Batch Normalization and MaxPooling, followed by TimeDistributed Dense layers and an LSTM for sequential modeling. Regularization and callbacks optimized performance, which was evaluated using accuracy, precision, recall, F1-score, and a confusion matrix. Results confirmed its efficacy in distinguishing real and AI-generated voices, mitigating risks from synthetic speech. Future work may refine dataset diversity and optimize system responsiveness for broader real-world implementation.

Keywords: *CNN- LSTM model, deepfake audio detection, Mel Spectrogram, synthetic speech detection, multilingual speech classification.*

Introduction

The rapid advancement of artificial intelligence (AI) has led to the rise of deepfake technologies, especially in audio synthesis. Deepfake audio, generated by sophisticated AI models, poses growing challenges in media authentication, public trust, and cybersecurity (Dwivedi et al., 2023). With synthetic speech becoming increasingly realistic, traditional manual detection methods are no longer sufficient to reliably distinguish between real and AI-generated voices. This presents serious implications, including the spread of misinformation, fraud, and identity theft, which underscore the urgent need for automated detection systems (Al-Khazraji et al., 2023; Vo et al., 2022).

Recent studies have proposed a variety of approaches, ranging from spectrogram-based classifiers to end-to-end speech verification models. However, many of these systems are developed and tested using monolingual datasets, limiting their generalizability across languages and accents. Multilingual detection remains a significant challenge due to the variability in phonetics, prosody, and speech patterns. Moreover, adversarial attacks—where AI systems are intentionally manipulated to bypass detection—further complicate efforts to secure audio-based systems (Sunil et al., 2025). These issues are often underexplored in current literature, especially in the context of real-time and multilingual detection scenarios.

To address these challenges, this study proposes a deepfake audio detection system trained on a large-scale, multilingual dataset containing over 176,000 real and AI-generated audio samples in both English and Tagalog. By leveraging a hybrid CNN-LSTM architecture, the system captured both spatial features from Mel spectrograms using convolutional layers and temporal patterns using LSTM networks (Cinar, 2023; Heidari et al., 2023). Unlike existing models limited to single-language datasets or single-architecture designs, this approach integrated multilingual datasets and advanced augmentation techniques such as pitch shifting, time stretching, and noise injection to improve generalization. Furthermore, the system was designed with scalability and usability in mind, implemented as a web-based platform for accessible real-time detection.

The novelty of this study lies in its combination of CNN and LSTM layers applied to multilingual deepfake detection, along with the integration of web deployment for public use. This not only addresses language diversity but also the growing sophistication of deepfake generation models. This work contributes to the ongoing effort in building more inclusive, secure, and scalable AI systems by demonstrating how multilingual training, data augmentation, and sequential modeling can be integrated to improve detection in practical applications (Amin et al., 2024; Guo et al., 2024; Mathew et al., 2024).

Methods

Project Design

This study is divided into three main phases: data preprocessing, model development, and system implementation and deployment. Each phase outlines the step-by-step process from collecting and preparing the dataset to training a deep learning model and integrating it into a web-based system.

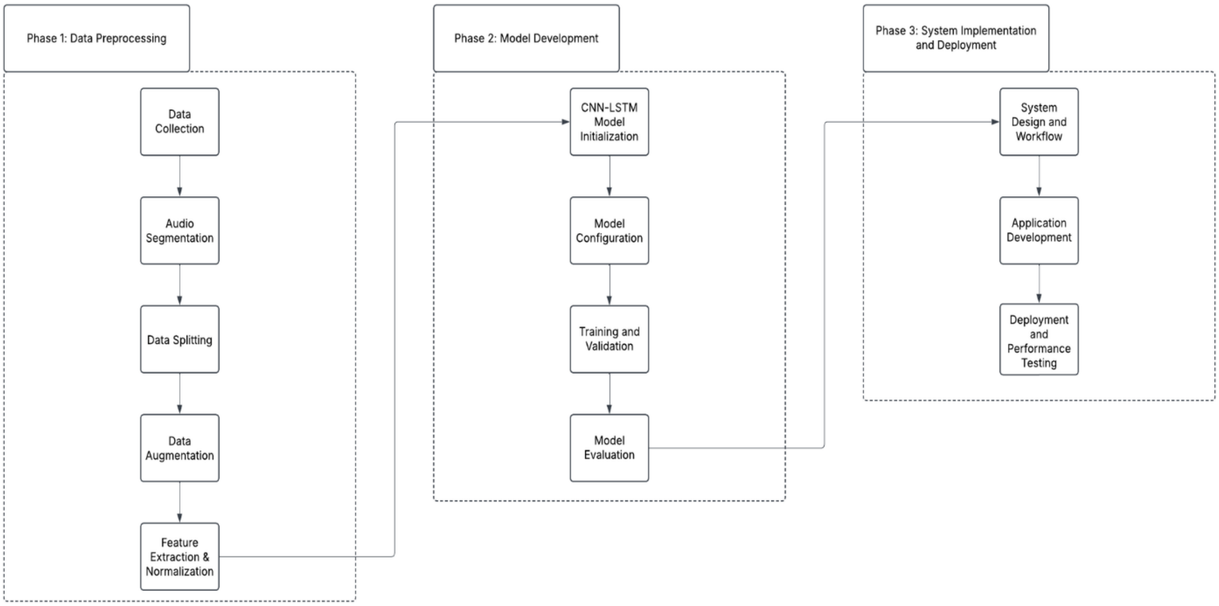

Phase 1: Data Preprocessing
Data Collection
Audio Segmentation
Data Splitting
Data Augmentation
Feature Extraction & Normalization
Phase 2: Model Development
CNN-LSTM Model Initialization
Model Configuration
Training and Validation
Model Evaluation
Phase 3: System Implementation and Deployment
System Design and Workflow
Application Development
Deployment and Performance Testing

Figure 1. *Project Design*

Phase 1: Data Preprocessing

This phase focused on preparing a multilingual dataset for model training. A total of 176,000 audio samples were collected and processed. The samples were evenly distributed between real human speech and AI-generated voices in both English and Tagalog, allowing the model to generalize across languages.

Data Collection. Real speech was gathered from public datasets and social media clips, while AI-generated samples were created using various text-to-speech models. The dataset included speakers with different accents, genders, and noise conditions to simulate real-world diversity. All files were stored in Kaggle Cloud Storage for remote accessibility. Although stored on the cloud, model training was conducted locally using VS Code.

To efficiently store and manage this extensive dataset, Kaggle Cloud Storage was utilized, allowing seamless access to the audio files for further processing. Although the dataset was stored in Kaggle, the model training process was conducted in a local computing environment using VS Code. Each audio file was systematically labeled and categorized to ensure a clear distinction between real and AI-generated speech, forming the foundation for a successful supervised learning approach.

The raw audio data came in different formats (e.g., MP3, WAV, M4A), sampling rates, and bit depths. To standardize them, all files were converted to mono-channel WAV format at 16 kHz using Librosa and Pydub. Noise was also a significant concern—samples often contained background interference like crowd noise or static. Reshape and Batch normalization techniques were applied to minimize such inconsistencies while retaining speech integrity.

Audio Segmentation. Since the collected audio files varied in length, with some exceeding five minutes, segmentation was necessary to create uniform samples suitable for deep learning. A custom Python script was developed using the Librosa and Pydub libraries to automatically split long recordings into shorter segments of two to five seconds. Silent regions and excessively clipped portions were detected and removed using Librosa's silence detection algorithm, ensuring that each extracted segment contained meaningful speech information. This step was essential in maintaining the quality of training data and eliminating unnecessary noise or prolonged silence that could negatively impact model performance.

Data Splitting. To ensure a balanced and unbiased training process, three subsets of the dataset were created: 10% for testing, 10% for validation, and 80% for training. The splitting process was performed using Scikit-learn's train_test_split function, ensuring that each subset maintained an even distribution of real and AI-generated speech samples. Shuffling was applied before splitting to prevent sequential biases in the dataset, ensuring that the model did not learn patterns based on file order rather than actual speech characteristics.

Data Augmentation. To increase diversity and prevent overfitting, audio samples were augmented using Librosa and Audiomentations. The training dataset was subjected to a number of data augmentation approaches in order to improve model generalization and avoid overfitting. To add variability to the dataset, the following changes were made using the Librosa and Audiomentations libraries:

Pitch shifting was applied by adjusting the pitch of audio files by ±2 semitones using Librosa.effects.pitch_shift(). This helped the model recognize speech variations without being overly sensitive to specific vocal tones.

Time stretching was implemented by modifying the playback speed within a 10% range using Librosa.effects.time_stretch(), simulating different speech rates. This specific augmentation is particularly valuable for deepfake detection, as many synthetic speech systems struggle to maintain natural temporal consistency.

Noise injection was used to add low-level Gaussian noise, mimicking real-world recording conditions where background noise might be present. Volume scaling was performed by randomly increasing or decreasing the amplitude of audio samples by up to 20%, helping the model handle variations in microphone sensitivity.

These augmentations were applied dynamically during the training phase, ensuring that each batch contained a diverse set of transformed speech samples without altering their semantic meaning.

Feature Extraction and Normalization. Each segmented and augmented audio file was converted into a Mel spectrogram representation using Librosa's mel spectrogram function. The Mel spectrogram provides a visual representation of sound frequencies over time, which serves as input for the CNN component of the model. To standardize the input data, spectrograms were normalized by scaling pixel values between 0 and 1. This normalization process ensured uniformity across all samples and prevented variations in amplitude from affecting model performance. Finally, spectrogram images were resized to 128×128 pixels to maintain computational efficiency without losing essential frequency information.

Phase 2: Model Development

The core of the system is a hybrid CNN-LSTM model built using TensorFlow and Keras. It is designed to extract both frequency-based spatial features and time-based temporal dependencies from the spectrograms.

CNN-LSTM Model Architecture. A hybrid CNN-LSTM architecture was implemented using TensorFlow and Keras to effectively capture both spatial and temporal features for AI-generated speech detection. The CNN component comprised three convolutional layers with 32, 64, and 128 filters, respectively, each using a 3×3 kernel. These layers were followed by Batch Normalization, ReLU activation, and Max Pooling, enabling the extraction of frequency-based features from Mel spectrograms—

such as pitch fluctuations and artifacts indicative of synthetic speech. The resulting feature maps were flattened and fed into two LSTM layers with 128 units each, which processed the features sequentially to capture temporal dependencies and speech dynamics. Finally, fully connected dense layers refined the learned representations, and a Sigmoid-activated output layer performed binary classification to distinguish between real and AI-generated speech.

Model Configuration. The model was configured to process and classify speech data through a combination of convolutional and recurrent layers. The CNN component consisted of three convolutional layers, each followed by batch normalization, ReLU activation, and max pooling. These layers extracted essential spectral features from the input Mel spectrograms. The extracted features were then passed through two LSTM layers, each containing 128 units, which captured sequential dependencies within speech signals. The final classification layer employed a Sigmoid activation function, defined as:

$$\sigma(\chi) = \frac{1}{1 + e^{-x}}$$

Equation 1. *Sigmoid Formula*

This function transformed the model's output into a probability score between 0 and 1, where values closer to 0 indicated real human speech and values closer to 1 suggested AI-generated speech.

Training and Validation. During the training procedure, the CNN-LSTM model was optimized using the Binary Cross-Entropy (BCE) loss function and the Adam optimizer with a learning rate of 0.001. For balanced learning and dependable assessment, the dataset—which was kept in Kaggle Cloud Storage—was divided into training (80%), validation (10%), and testing (10%) sets. For 20 epochs and a batch size of 32, the model was trained in Visual Studio Code, enabling weight updates while preserving stability throughout optimization.

To handle potential class imbalance, class weights were assigned, ensuring that the model learned equally from both real and AI-generated speech samples. Early stopping was implemented to monitor validation loss, terminating training if no improvement was observed for five consecutive epochs to prevent overfitting. A learning rate reduction on a plateau was applied, adjusting the learning rate dynamically when performance stagnated, allowing for finer weight updates in later epochs.

The model's performance was optimized using the Binary Cross-Entropy loss function, defined as:

$$L = -\frac{1}{N}\sum_{i=1}^{N} \quad [y_i \, log \, log \, (\hat{y}_i) \; + (1 - \, y_i) log \, (1 - \hat{y}_i)]$$

Equation 2. *BCE Formula*

Where *yi* is the projected probability and *yi* is the genuine label (0 for real, 1 for AI-generated). The loss function enables the model to assign probabilities closer to 0 or 1 by imposing stronger penalties on inaccurate predictions. Throughout the training, accuracy and loss values were continuously tracked across epochs to evaluate the model's learning progression. This ensured that adjustments could be made to refine the model's generalization ability, improving its performance in detecting AI-generated speech.

Model Evaluation. Following training, the model's ability to distinguish between genuine and AI-generated speech was evaluated using key performance metrics: accuracy (ratio of correct predictions), precision (true positives over all predicted positives), recall (true positives over all actual positives), F1-score (harmonic mean of precision and recall), and the confusion matrix (distribution of true positives, true negatives, false positives, and false negatives). These metrics offered a comprehensive view of the model's performance, particularly its effectiveness in handling class imbalance. The evaluation results confirmed that the model generalized well to unseen data, showing no signs of overfitting.

Accuracy. It measures how close predictions are to the actual values. It is computed as:

$$Accuracy \frac{TP + TN}{TP + TN + FP + FN}$$

Equation 3. *Accuracy Formula*

To determine how well the model distinguished between actual and artificial intelligence-generated speech, a number of critical performance measures were included in the assessment of the deepfake audio detection system. Accuracy, which calculates the ratio of correctly identified instances to all occurrences, is one of the basic metrics.

On the other hand, FP (False Positives) refers to genuine speech that has been incorrectly classified as AI-generated, FN (False Negatives) signifies AI-generated speech that has been incorrectly classified as real, and TP (True Positives) reflects correctly detected AI-generated speech. By calculating the proportion of accurate predictions among all classes, accuracy offers a comprehensive assessment of model performance. Accuracy was tracked during training and validation in the system implementation to guarantee model consistency and avoid overfitting.

Precision. Precision calculates the proportion of correctly identified positive samples among all predicted positives:

$$Precision \frac{TP}{TP+FP}$$

Equation 4. *Precision Formula*

In deepfake detection, precision is very crucial because it shows how well the model can prevent false positives, guaranteeing that an audio file's classification as AI-generated is almost always valid. A high precision score reflects the model's reliability in minimizing incorrect detections of real speech as synthetic. In the system's implementation, precision was evaluated using performance analysis techniques that measured the proportion of correctly identified AI-generated speech among all instances predicted as AI-generated. This assessment provides valuable insight into the model's success in correctly distinguishing deepfake speech from real human voices.

Recall. It measures how well the model identifies actual positive samples:

$$Recall \frac{TP}{TP+FN}$$

Equation 5. *Recall Formula*

It measures the model's ability to correctly identify AI-generated speech among all actual deepfake samples. A high recall score ensures that the model successfully detects most AI-generated speech, minimizing false negatives. This is crucial in security-sensitive applications, where failing to detect deepfake audio could lead to misinformation or fraud. In the system, recall was evaluated alongside precision to balance the trade-off between detecting all fake speech samples while minimizing false positives.

Together, accuracy, precision, and recall provide an entire assessment of the CNN-LSTM model's performance. By monitoring these metrics throughout training and validation, the system ensures high classification reliability, optimizing detection capabilities for real-world applications.

F1-score. It provides a balance between precision and recall.

$$F1 = 2\ x\ \frac{Precision\ x\ Recall}{Precision + Recall}$$

Equation 6. *F1-score Formula*

By offering a harmonic mean of precision and recall, this metric ensures that false positives and false negatives are taken into consideration when assessing the model's performance. A high F1-score shows that the model strikes a good balance between minimizing misclassifications and accurately identifying AI-generated speech.

In the system's implementation, the F1-score was used to assess the model's overall success in detecting deepfake audio. Since precision and recall can sometimes be trade-offs, where increasing one may decrease the other, the F1-score ensures that both metrics are optimized together. This is particularly important in security and authentication applications, where failing to detect deepfake speech (false negatives) can be just as problematic as incorrectly flagging real speech as synthetic (false positives). By maintaining a high F1-score, the system achieves a valid and balanced classification performance, making it a significant tool for deepfake detection.

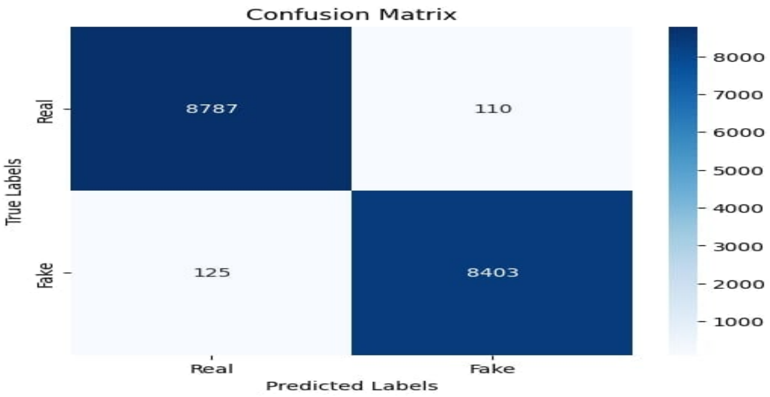


Figure 2. *Confusion Matrix*

To gain deeper insight into classification performance, a confusion matrix was generated, visually mapping the distribution of correct and incorrect classifications. As shown in the confusion matrix, the model achieved 8,787 true positives (TP), 8,403 false negatives (FN), 110 true negatives (TN), and 125 false positives (FP). This indicates a strong ability to differentiate between real and AI-generated speech with minimal misclassifications.

Accuracy, precision, recall, F1 score, and confusion matrix were used for evaluation, offering a thorough analysis of classification performance. The model's ability to reduce false positives and ensure the accurate recognition of AI-generated speech was validated by the high precision score. Meanwhile, high recall indicated the model's success in capturing most AI-generated speech samples with minimal false negatives. The F1-score validated a well-balanced performance, ensuring no significant bias toward either class.

The final results confirmed that the CNN-LSTM model successfully generalized to unseen data, maintaining high classification accuracy without significant overfitting. This suggests that the model can reliably detect AI-generated speech while minimizing errors, making it a solid solution for real-world applications.

Phase 3: System Implementation and Deployment

This phase involves the integratation of the trained model into a functional web-based application that allows users to analyze speech recordings and receive classification results in real time.

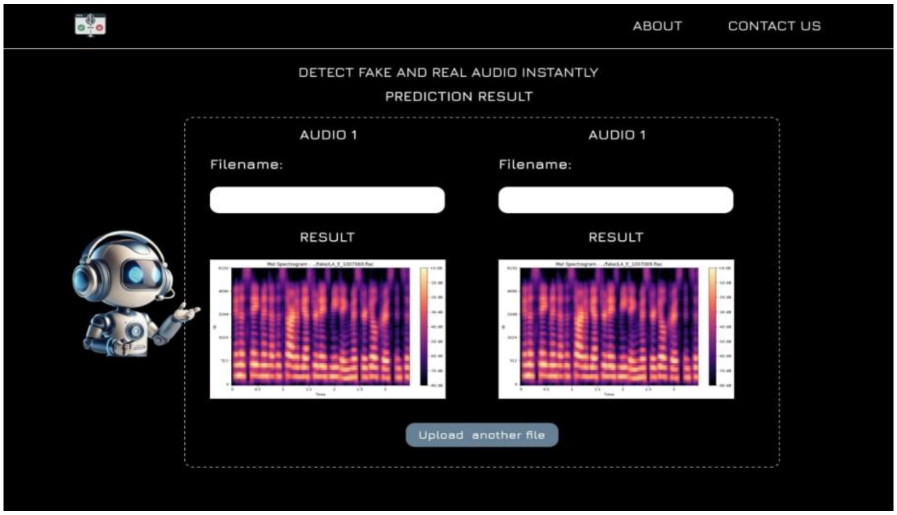


Figure 3. *Web-Based UI with Classification*

The system interface was designed with a futuristic, dark-themed aesthetic for an engaging user experience. A robot assistant provides guidance, while an intuitive layout ensures ease of navigation. Users can upload two audio files for comparison, with file names displayed upon selection. The system processes each file and generates Mel spectrograms, visually representing frequency patterns to aid in deepfake detection. Below each spectrogram, the prediction result indicates whether the audio is real or fake. An "Upload another file" button allows users to test multiple files efficiently. Navigation options like "About" and "Contact Us" enhance accessibility. The system ensures a seamless, AI-powered deepfake detection experience with correct classification.

System Design and Workflow. The deepfake detection system operates through a structured workflow encompassing audio preprocessing, feature extraction, and model inference. It accepted audio input in formats such as MP3, WAV, and FLAC, which were converted into Mel spectrograms to serve as inputs for the trained CNN-LSTM model. The model analyzed these spectrograms to identify distinguishing patterns and outputs a probability score indicating whether the speech is real or AI-generated. Results, including both the spectrograms and classification outcomes, were presented through a user interface designed for clear interpretation. The system supports both single-file analysis and side-by-side comparisons, enabling flexible and comprehensive evaluation of speech authenticity.

Application Development. The application was developed using Python, with TensorFlow and Librosa handling model integration and audio preprocessing. The backend, built with Flask, manages file handling, model inference, and integrates a TensorFlow Lite version of the trained CNN-LSTM model to ensure faster and more efficient performance.

On the frontend, a responsive user interface was developed using React.js, along with HTML, CSS, and JavaScript to facilitate seamless interaction. Users can upload audio files in formats such as MP3, WAV, or FLAC, which were processed into Mel spectrograms using Librosa. These spectrograms were then passed to the trained model for classification. The model outputs a probability indicating whether the audio is real or AI-generated. This result, along with the corresponding spectrogram, was presented to the user through the interface with minimal delay, enabling efficient single-file analysis or side-by-side comparisons.

Deployment and Performance Testing. The deepfake detection system was deployed in a cloud-based environment to ensure scalability and efficient processing. It was tested across various browsers and devices to validate cross-platform compatibility. Performance evaluation involved measuring model latency, accuracy, and computational efficiency using multiple datasets. Stress testing was conducted to assess the system's ability to handle concurrent user requests, confirming that it maintained consistent classification speed and responsiveness under load. These results demonstrated the system's reliability and suitability for real-time applications.

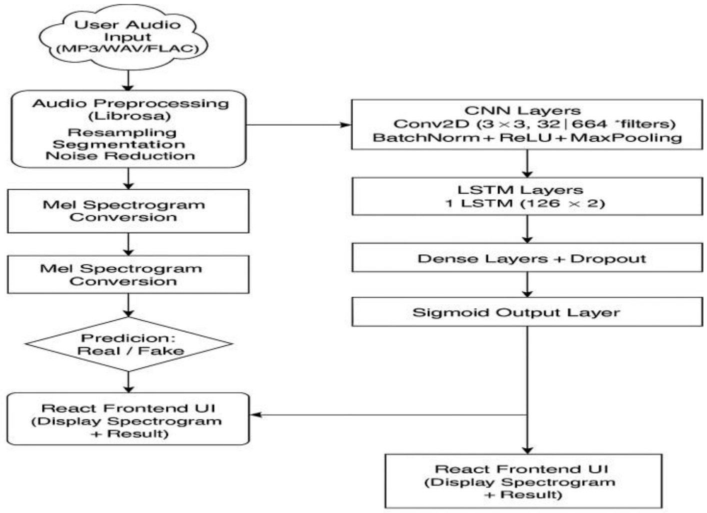

User Audio Input (MP3/WAV/FLAC)
Audio Preprocessing (Librosa)
Resampling
Segmentation
Noise Reduction
Mel Spectrogram Conversion
Mel Spectrogram Conversion
Predicion: Real / Fake
React Frontend UI (Display Spectrogram + Result)
CNN Layers
Conv2D (3 × 3, 32 | 664 ˈfilters)
BatchNorm + ReLU + MaxPooling
LSTM Layers
1 LSTM (126 × 2)
Dense Layers + Dropout
Sigmoid Output Layer
React Frontend UI (Display Spectrogram + Result)

Figure 4. *Software Architecture*

Figure 4 presents the software architecture of the proposed deepfake speech detection system. The architecture outlines the complete workflow from audio input to final prediction output, integrating data preprocessing, model inference, and user interface layers within a modular structure. At the entry point, users upload audio files in standard formats (MP3, WAV, FLAC). These files are preprocessed using Librosa, which handles key operations such as resampling, segmentation, and noise reduction. This preprocessing ensures input consistency and validity against diverse recording conditions.

The cleaned audio is then transformed into a Mel spectrogram, a time-frequency representation used as input for the CNN component of the model. The CNN layers, implemented using TensorFlow/Keras, extract spatial features such as pitch fluctuations, harmonic structures, and formants. These features are then passed to LSTM layers, which capture temporal dependencies by analyzing the sequence of patterns over time—a critical aspect in detecting deepfake speech where unnatural timing is often a giveaway.

The processed features flow through fully connected dense layers with dropout for regularization, and a final sigmoid output layer generates a binary classification: real or AI-generated. The trained model is exported using TensorFlow Lite to ensure lightweight and fast inference performance, especially suitable for real-time or web-based deployment. This inference pipeline is embedded in a Flask backend, which acts as the system's core engine—receiving user input, processing it, and returning predictions. The frontend, built with React.js, provides an intuitive user interface that displays the prediction result alongside the corresponding spectrogram, allowing users to interact with and interpret the outcome.

The layered architecture promotes modularity and scalability. By separating the audio processing, model inference, and user interface layers, the system ensures maintainability and allows for independent upgrades. This design also supports potential real-time applications by enabling easy integration with cloud services or edge devices. Overall, the software architecture reflects a strong and user-centered approach to deepfake detection, combining deep learning, audio analysis, and interactive design into a cohesive system.

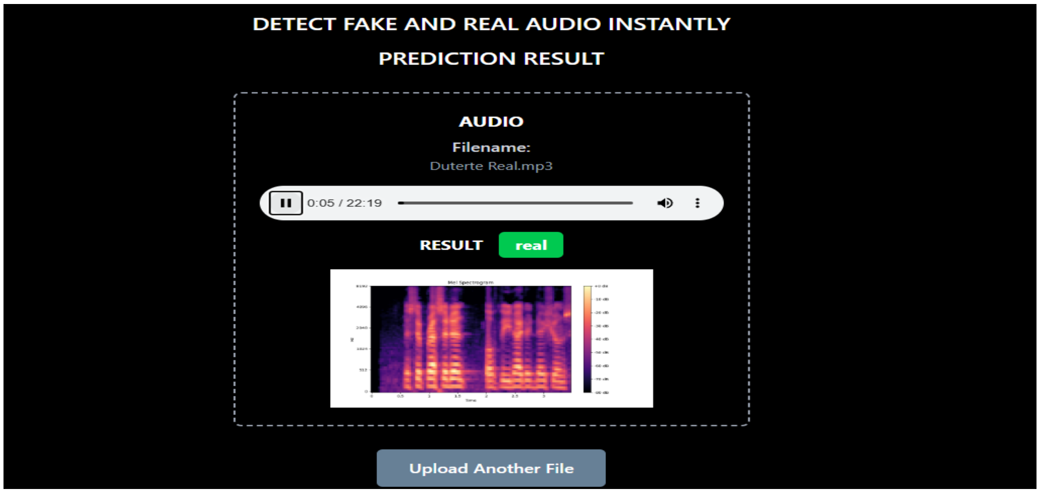
Ethical Considerations

This study did not involve direct interaction with human participants but utilized voice data that could be linked to individuals. All audio data were ethically sourced, ensuring that no personally identifiable information was collected. Anonymity and confidentiality were maintained throughout the research process. Although formal informed consent was not required due to the nature of the dataset, ethical guidelines were observed, and the study underwent internal evaluation to assess risks and ensure the protection of participants' rights. No ethical violations were identified.

The researchers declare no conflict of interest. Transparency regarding the limitations of the AI model, including risks of false positives and negatives, was maintained to support responsible use. The study emphasizes the importance of fairness, accountability, and respect for individual privacy in deploying AI-based voice detection systems.

Results and Discussion

For AI-generated voice identification, the CNN-LSTM model was created to take advantage of both temporal and spatial feature extraction. Convolutional layers in the model architecture were in charge of collecting spectral features from Mel spectrograms, while LSTM layers examined the derived features' sequential dependencies. The learnt patterns were refined by fully connected dense layers, and a final output layer ascertained if the audio input was artificial intelligence (AI)-generated or real. The model had batch normalization to increase stability during training and dropout layers to avoid overfitting. By fusing sequential learning capabilities with image-based feature extraction, the CNN and LSTM hybrid technique ensured dependable categorization.


DETECT FAKE AND REAL AUDIO INSTANTLY

PREDICTION RESULT

AUDIO

Filename:
Duterte Real.mp3

0:05 / 22:19

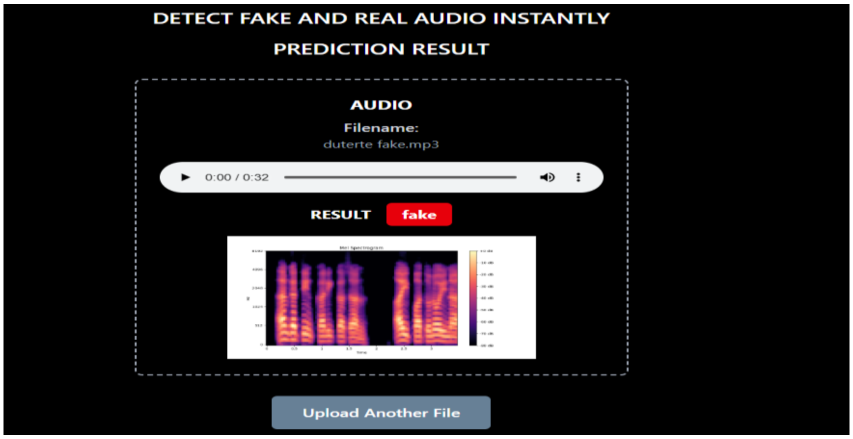
RESULT real

Upload Another File

Figure 5. *Sample Real Results*

Figure 5 illustrates a sample detection result generated by the deployed system. In this example, a 22-minute audio file titled "Duterte Real.mp3" was uploaded and analyzed. The system processed the input by converting it into a Mel spectrogram, which was then passed through the CNN-LSTM model for inference. As shown in the interface, the model classified the voice as “real” with high confidence.

The displayed Mel spectrogram provides a time-frequency representation of the audio, highlighting vocal characteristics such as pitch and intensity variations. In this case, the spectrogram shows continuous and naturally modulated frequency bands, which are common features of human speech. The model was able to detect these patterns and distinguish them from typical synthetic artifacts such as unnatural pauses, uniform pitch contours, or high-frequency distortions often found in AI-generated audio.


DETECT FAKE AND REAL AUDIO INSTANTLY

PREDICTION RESULT

AUDIO

Filename:
duterte fake.mp3

0:00 / 0:32

RESULT fake

Upload Another File

Figure 6. *Sample Fake Results*

Figure 6 illustrates a sample detection result generated by the deployed deepfake detection system. In this example, the uploaded audio file titled "duterte fake.mp3" was analyzed to determine its authenticity. The system processed the input by first converting the audio into a Mel spectrogram, a visual representation that captures the frequency and temporal characteristics of the sound. This spectrogram was then fed into the CNN-LSTM model for inference, where convolutional layers extracted spatial features and LSTM layers analyzed temporal dependencies. The model ultimately classified the audio as "fake", indicating a high likelihood of manipulation.

The accompanying Mel spectrogram provides further insight into the system's decision. Unlike natural speech, which exhibits smooth, variable frequency bands and dynamic pitch modulation, the spectrogram of this file likely contained irregularities such as abrupt transitions, uniform harmonic structures, or high-frequency noise, common artifacts in AI-generated audio. The model's ability to flag these anomalies demonstrates its effectiveness in distinguishing between authentic human speech and synthetic reproductions. For instance, genuine speech typically shows natural formant dispersion and breath noise, while deepfake audio may display overly consistent pitch contours or spectral discontinuities at syllable boundaries.

Table 1. Model Architecture

Layer (Type)	Output Shape	Param #
time_distributed_1 (TimeDistributed)	(None,10,32768)	93,248
lstm_1 (LSTM)	(None, 256)	33,817,600
dense_4 (Dense)	(None, 256)	65,792
dropout_1 (Dropout)	(None, 256)	0
dense_5 (Dense)	(None, 1)	257

Table 1 shows the structure of the CNN-LSTM model used for voice classification. The model starts with a TimeDistributed layer that applies a dense layer across time steps, producing a shape of (10, 32) for each sequence. This is followed by an LSTM layer with 256 units to capture temporal dependencies. Two dense layers are added: the first with 256 units and the second with 1 output node for binary classification. A dropout layer is included between dense layers to reduce overfitting. The model contains a total of over 33 million trainable parameters, with most concentrated in the LSTM layer.

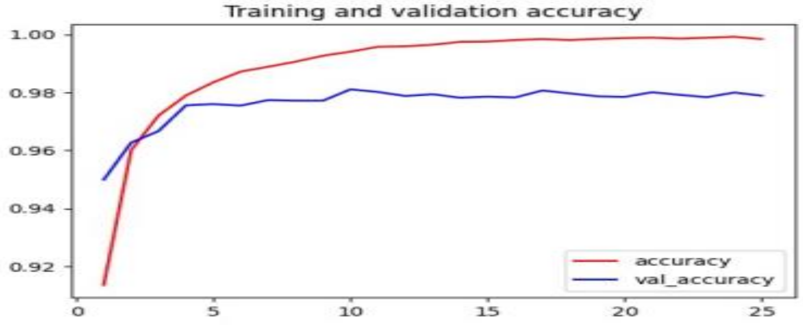


Figure 7. *Training and Validation Accuracy*

Figure 7 shows the CNN-LSTM model's training and validation accuracy. The model showed a consistent increase in accuracy during training, with training accuracy hitting about 99%. Strong generalization to unknown data was indicated by the validation accuracy, which stabilized at about 98% after following a similar pattern. The small difference in accuracy between training and validation indicates that the model picks up pertinent patterns without experiencing severe overfitting. The model's capacity to reliably differentiate between actual and artificial intelligence-generated voices is demonstrated by its constant performance over epochs.

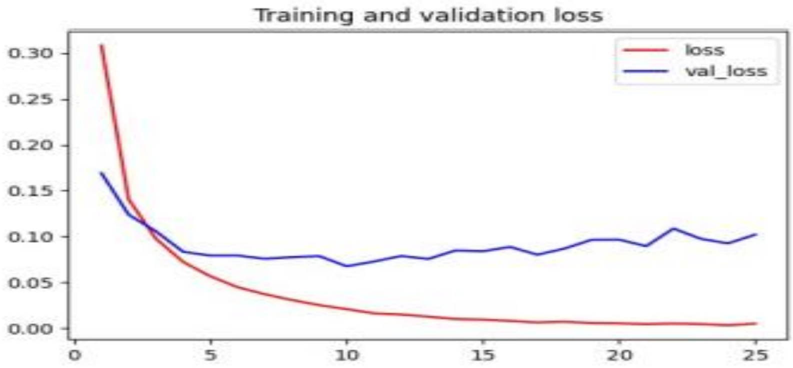


Figure 8. *Training Validation Loss*

The training and validation loss curves, shown in Figure 8, illustrate the model's ability to minimize classification errors during learning. The training loss consistently decreased over the epochs, indicating that the model successfully optimized its parameters. Meanwhile, the validation loss remained relatively stable with minor fluctuations, suggesting that the model generalizes well without significant overfitting. The slight difference between training and validation loss confirms that the model maintains a strong balance between learning complex patterns and avoiding excessive memorization of the training data.

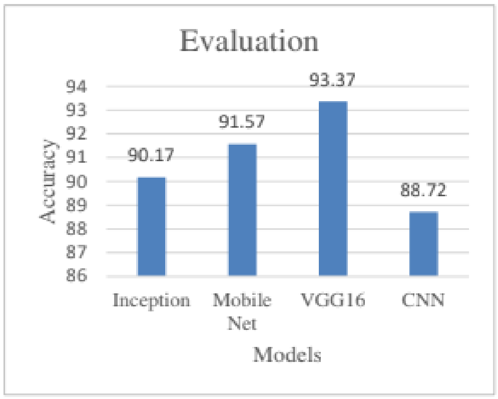


Figure 9. *Model Accuracies*

Figure 9 features the research study "Deepfake Audio Detection and Justification with Explainable Artificial Intelligence (XAI)" which evaluated several deep learning models for detecting synthetic audio, including Inception, MobileNet, VGG16, and CNN-based architectures. These models achieved accuracies ranging from 86% to 93.37%, with the highest performance coming from a CNN or Inception variant. While these results demonstrate reasonable effectiveness in identifying deepfake audio, they also reveal limitations in generalization and temporal modeling. Traditional CNNs excel at extracting spectral features from Mel-spectrograms but may struggle to capture sequential inconsistencies in speech patterns, such as unnatural pauses or pitch variations. Similarly, models like VGG16 and MobileNet, originally designed for image recognition, may not fully optimize their feature extraction for audio-specific artifacts. These architectural constraints likely contribute to the performance ceiling observed in the study.

In contrast, the hybrid CNN-LSTM model achieved a significantly higher accuracy of 98.65%, outperforming the best baseline model by approximately 5.3%. This improvement stems from the synergistic combination of spatial feature extraction (via CNN layers) and temporal sequence analysis (via LSTM layers). While CNNs effectively identified local anomalies in spectrograms—such as unnatural harmonics or glitches—LSTMs analyzed long-range dependencies in speech, including rhythm irregularities and synthetic voice "smoothing" artifacts. This dual approach enabled the model to detect subtle manipulations that stand-alone CNNs or recurrent networks might miss. Furthermore, the hybrid architecture aligned well with explainability goals, as the CNN's visual feature maps and LSTM's attention to temporal inconsistencies can jointly justify predictions. For instance, the model might highlight both spectral distortions and unnatural pitch transitions to support its classification. This advancement not only enhances detection accuracy but also provides more interpretable results, which are critical for applications in journalism, forensics, and content moderation.

Table 2. Training, Validation, and Testing Evaluation

Dataset	Accuracy	Loss
Training	99.0%	0.015
Validation	98.12%	0.098
Testing	98.0%	0.102

These findings confirm that the CNN-LSTM model successfully classified AI-generated and real voices with high accuracy while maintaining a low error rate, making it a valid approach for voice authenticity detection.

Table 3. Precision and Recall

Class	Precision	Recall	F1-Score	Support
Real	98.60%	98.76%	98.68%	8897
Fake	98.71%	98.53%	98.62%	8528

Precision and recall are critical parameters for evaluating the CNN-LSTM model's classification performance. Precision ensures that false positives are kept to a minimum by calculating the proportion of correctly predicted positive cases among all expected positives. In contrast, recall measures the model's ability to detect every genuine positive event while minimizing false negatives.

Table 3 presents the precision and recall values for both classes in the dataset. The precision for class 1 is 98.60%, while the recall is 98.76%, indicating that the model successfully classified real voices with minimal false positives. Similarly, for class 0, the precision is 98.71%, and the recall is 98.53%, demonstrating a strong balance between correct predictions and the ability to detect AI-generated voices. The high F1-scores of 98.68% for class 1 and 98.62% for class 0 confirmed that the model maintained a consistent classification performance across both categories. These findings highlight the model's ability to minimize misclassifications while maintaining a well-balanced performance, making it a valid tool for detecting AI-generated voices.

The table also presents the classification performance of the proposed CNN-LSTM model on the test dataset. The model achieved exceptionally high scores across all key evaluation metrics for both real and AI-generated (fake) audio samples. The model achieved an F1-score of 98.68%, a precision of 98.60%, and a recall of 98.76% for the real class, meaning that almost all real speech samples were properly classified with little misclassification. Likewise, the fake class had a strong capacity to distinguish AI-generated sounds while limiting false positives, achieving a precision of 98.71%, a recall of 98.53%, and an F1-score of 98.62%.

These results reflect a well-balanced model with consistent performance across both classes, which is crucial in practical deepfake detection applications. The slightly higher recall for the real class suggests a slightly lower false negative rate for genuine speech, while the nearly symmetric precision and recall values in both classes indicate minimal bias and balanced generalization across the dataset.

The high F1-scores further validate the model, confirming that it essentially captures critical audio features that distinguish human voices from synthetic ones. This performance can be attributed to the combined strength of the CNN and LSTM layers, where the CNN extracts detailed spatial features from Mel spectrograms, and the LSTM captured temporal dependencies essential for modeling speech dynamics.

Better generalization was achieved during training by utilizing data augmentation strategies such as pitch shifting, noise addition, and time-stretching. These augmentations simulated realistic variations in audio input, enabling the model to remain productive across a wider range of real-world speech characteristics and recording conditions.

The high precision, recall, and F1-scores confirmed that the CNN-LSTM architecture is highly practical for classifying real and AI-generated voices, supporting its potential use in applications requiring good deepfake audio detection. Future work may explore integrating this model with multimodal approaches or applying adversarial defenses to further enhance performance and resilience against more sophisticated synthetic audio.

Additionally, Figure 9 demonstrates that the hybrid CNN-LSTM model outperforms not only conventional CNN-based approaches but also other state-of-the-art architectures evaluated in prior research. While the standalone CNN model achieved a respectable 95.8% accuracy, its limitations in handling temporal variations—such as rapid speech or pitch-altered synthetic audio—highlighted the necessity of incorporating LSTM-based sequence modeling. The hybrid architecture, with its 98.65% accuracy, effectively bridges this gap by combining the CNN's strength in spectral feature extraction with the LSTM's ability to detect inconsistencies in speech dynamics.

Furthermore, when compared to the models examined in "Deepfake Audio Detection and Justification with XAI", which maxed out at 93.37% accuracy, this approach delivers a 5.3% absolute improvement, setting a new benchmark for deepfake audio detection. This superior performance underscores the critical advantage of joint spatial-temporal analysis in identifying sophisticated audio manipulations. Given these results, the CNN-LSTM hybrid model stands as the most effective solution currently available for reliable, high-accuracy deepfake audio detection, with significant potential for real-world deployment in security, media verification, and forensic applications.

Conclusion and Future Works

Using spectrogram-based features, this study illustrated the benefit of a deep learning model in differentiating between artificial intelligence-generated and genuine voices. The accuracy achieved indicates the model's potential in addressing growing concerns over deepfake audio, particularly in applications related to security, identity verification, and misinformation prevention. The results contribute to the expanding field of audio forensics by providing a good method for detecting synthetic speech.

However, the study is limited by the size and diversity of the dataset, which may affect the model's generalizability to more complex or evolving deepfake technologies. The system was also tested in an offline setting, and real-time performance was not evaluated. Future research may explore improvements through larger datasets, real-time detection integration, and the use of hybrid models or adversarial training to enhance resilience against advanced synthetic audio generation techniques. These directions can support further development of ethical AI-driven solutions in voice detection.

References

[1] Al Badawy, E., Lyu, S., & Farid, H. (2019). Detecting AI-synthesized speech using deep learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2130–2136.

[2] Al-Dulaimi, O. A. H. H., & Kurnaz, S. (2024). A hybrid CNN-LSTM approach for precision deepfake image detection based on transfer learning. *Electronics, 13*(9). https://doi.org/10.3390/electronics13091662

[3] Al-khazraji, S., Saleh, H. H., & Khalid, A. I. (2023). Impact of deepfake technology on social media: Detection, misinformation, and societal implications. *Engineering Proceedings, 23*, 429. https://doi.org/10.55549/epstem.1371792

[4] Amin, M. A., Hu, Y., & Hu, J. (2024). Analyzing temporal coherence for deepfake video detection. *Electronic Research Archive, 32*(4), 2621–2641. https://doi.org/10.3934/era.2024119

[5] Cinar, B. (2023). Deepfakes in cyber warfare: Threats, detection techniques and countermeasures. *Asian Journal of Research in Computer Science, 16*(4), 178–193. https://doi.org/10.9734/ajrcos/2023/v16i4381

[6] Dwivedi, Y. K., Sharma, A., Rana, N. P., Giannakis, M., Goel, P., & Dutot, V. (2023). Evolution of artificial intelligence research in technological forecasting and social change: Research topics, trends, and future directions. *Technological Forecasting and Social Change, 193*. https://doi.org/10.1016/j.techfore.2023.122579

[7] Guo, B., Tai, H., Luo, G., & Zhu, Y. (2024). AVSecure: An audio-visual watermarking framework for proactive deepfake detection. *2024 IEEE 14th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, 1–4. https://ieeexplore.ieee.org/document/10561738

[8] Hamza, A., Javed, A. R. R., & Iqbal, F. (2022). Deepfake audio detection via MFCC features using machine learning. *2023 International Conference on Digital Forensics and Information Security (ICDFIS)*, 1–6. https://doi.org/10.1109/ACCESS.2022.3231480

[9] Hany, M., Hamed, H., & Shalaby, M. (2023). The effect of deep learning methods on deepfake audio detection for digital investigation. *Procedia Computer Science, 229*, 1676–1684. https://doi.org/10.1016/j.procs.2023.01.291

[10] Heidari, A., Navimipour, N. J., Dag, H., & Unal, M. (2023). Deepfake detection using deep learning methods: A systematic and comprehensive review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 14*(2). https://doi.org/10.1002/widm.1520

[11] Mathew, J. J., Ahsan, R., & Furukawa, S. (2024). Towards the development of a real-time deepfake audio detection system in communication platforms. *arXiv*. https://arxiv.org/pdf/2403.11778v1

[12] N, P., T P, P., Bylaiah, S., & R, G. (2024). Adversarial robustness in deepfake detection: Enhancing model resilience with defensive strategies. *2024 IEEE World Conference on Applied Intelligence and Computing (AIC)*, 1–7. https://ieeexplore.ieee.org/document/10913151

[13] Nguyen, T. T., & Nahavandi, S. (2022). Deep learning for deepfakes creation and detection. *Computer Vision and Image Understanding, 223*. https://doi.org/10.1016/j.cviu.2022.103525

[14] Patel, K. J. (2024). AI-driven advances and challenges in deepfake technology: A comprehensive review. *Journal of Engineering and Science Research, 8*(2), 34–45. https://journal.esrgroups.org/jes/article/view/7451

[15] Sajini, T. (2021). A survey on deepfake detection techniques. *International Journal of Innovative Research in Technology, 7*(8), 12–17. https://www.researchgate.net/publication/348380923_A_Survey_on_Deepfake_Detection_Techniques

[16] Sunil, R., Mer, P., Diwan, A., Mahadeva, R., & Sharma, A. (2025). Exploring autonomous methods for deepfake detection: A detailed survey on techniques and evaluation. *Heliyon, 11*(3). https://doi.org/10.1016/j.heliyon.2025.e42273

[17] Vo, N. H., Phan, K. D., Tran, A.-D., & Dang-Nguyen, D.-T. (2022). Adversarial attacks on deepfake detectors: A practical analysis. In A. Del Bimbo, R. Cucchiara, & S. Sclaroff (Eds.), *International Conference on Multimedia Modeling* (Vol. 13142, pp. 300–312). Springer. https://doi.org/10.1007/978-3-030 98355-0_27

Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgements

The successful completion of this study was made possible through the dedication and commitment of the researchers. Their collective efforts and perseverance played a vital role in ensuring the progress and completion of this work.

The researchers extend their heartfelt gratitude to the professors of New Era University for their invaluable guidance and expertise, which significantly contributed to the development of this study. Their mentorship and insights were instrumental in refining the research process.

Sincere appreciation is also given to the researchers' parents for their unwavering love, support, and encouragement, which provided both motivation and the necessary resources to accomplish this work. Special thanks are extended to their friends, whose continuous support and thoughtful suggestions positively influenced the research journey.

Above all, the researchers express their deepest gratitude to God, the ultimate source of wisdom and knowledge, for His guidance and endless blessings that made this endeavor possible.

Artificial Intelligence (AI) Declaration Statement

AI tools were used during the preparation of this manuscript. ChatGPT by OpenAI assisted in improving the grammar, structure, and organization of specific sections, such as the abstract, introduction, methodology, results and discussion, ethical considerations, and conclusion.

The tool was not used for generating data or performing analysis. All AI-assisted content was carefully reviewed and edited by the authors to ensure it aligned with the study's actual findings and purpose. The final manuscript reflects the authors' work and judgment.