Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

June 2026. Vol. 5. No. 3; e-ISSN: 2808-4519

Voice Deepfake Detection Using Spectrogram Features and Mel-Frequency Cepstral Coefficients with a Combination of CNN and RNN

Ade Setiawan[1]*, Hermawan Syahputra[2], Yulita Molliq Rangkuti[3], Zulfahmi Indra[4], Kana Saputra S[5]

[1,2,3,4,5]Departement of Computer Science, Faculty of Mathematics and Natural Sciences, Universitas Negeri Medan, Indonesia
adee08setiawan@gmai.com[1]*

Abstract

The rapid development of artificial intelligence technology has driven the emergence of voice *deepfakes* that are increasingly realistic and difficult to distinguish from genuine human speech. This condition creates significant risks, including identity misuse, fraud, and information manipulation, thereby requiring an effective detection system capable of identifying manipulated voice patterns with high accuracy. This study aims to develop a voice *deepfake* detection system by combining *Convolutional Neural Network* (CNN) and *Recurrent Neural Network* (BiLSTM) methods. Audio features were extracted using *Mel-Frequency Cepstral Coefficients* (MFCC) and *spectrograms* to capture both frequency characteristics and temporal dynamics simultaneously. All genuine and *deepfake* voice recordings underwent preprocessing stages, including noise reduction, normalization, segmentation, and augmentation, to improve data diversity and model robustness. The model was evaluated using several data split ratios to determine the most optimal performance. The best result was achieved with an 80:20 ratio, reaching an accuracy of 99.3% and an AUC value of 0.9999. These results demonstrate the model's strong capability to identify subtle structural changes in audio signals that are difficult to detect using conventional methods. Based on these findings, the CNN–RNN (BiLSTM) approach proved to be highly effective for detecting manipulated voice recordings. This research provides an important contribution to the development of audio security systems and the mitigation of risks associated with the misuse of *deepfake* technology across various sectors.

Keywords: *Voice deepfake, Audio detection, Convolutional Neural Network, Recurrent Neural Network, Mel-Frequency Cepstral Coefficients, Spectrogram*

1. Introduction

The rapid development of *Artificial Intelligence* (AI) has significantly transformed human life by simplifying digital activities and improving efficiency in various sectors. However, the growth of AI also introduces serious challenges, including privacy violations, misinformation, algorithmic bias, and misuse of automated systems [1]. One of the most concerning consequences is the emergence of *deepfake* technology, which enables the manipulation of images, videos, and human voices into highly realistic synthetic media [2].

Among the various forms of *deepfake*, voice manipulation has become one of the most dangerous because it is difficult to detect manually and can be used for identity theft, fraud, misinformation, and financial crime [3]. Voice *deepfake* allows criminals to imitate executives, public figures, or trusted individuals to deceive victims into transferring money or disclosing sensitive information. A well-known case reported by Stupp involved fraudsters using AI-generated voice cloning to imitate a company director and successfully convince a CEO to transfer €220,000 to a fake supplier account [4]. Similar incidents have shown that victims often fail to distinguish genuine speech from manipulated audio because synthetic voices can sound highly convincing. The spread of *deepfake* content has increased rapidly worldwide. Data from the Ministry of Communication and Informatics of Indonesia reported that more than 95,000 *deepfake* videos were circulating globally, with a 550% increase since 2019 [5]. Surveys also show that one in four adults has experienced or knows someone affected by voice cloning fraud or *voice phishing* (*vishing*), while many users admit they cannot reliably differentiate real voices from AI-generated ones [6]. This situation proves that voice *deepfake* has become a serious cybersecurity threat requiring stronger detection mechanisms.

Although *deepfake* generation technology has advanced rapidly, detection systems still face major limitations. Public benchmark datasets such as the *DeepFake Detection Challenge* (DFDC) show that detection accuracy remains insufficient for real-world implementation [7]. Many previous studies focused mainly on visual-based detection, while audio-based *deepfake* detection remains relatively limited [8].

Some studies have proposed MFCC-based machine learning models and *Explainable Artificial Intelligence* (XAI) approaches, but these methods often rely heavily on static features and fail to fully capture temporal manipulation patterns in speech signals [9].

Since speech is naturally sequential and dynamic, effective detection requires both spatial and temporal feature learning. Therefore, this study proposes a hybrid approach using *Convolutional Neural Network* (CNN) and *Recurrent Neural Network* (RNN), specifically BiLSTM, combined with *Mel-Frequency Cepstral Coefficients* (MFCC) and *spectrogram* feature extraction. This method is expected to improve detection accuracy by learning both frequency structures and long-term temporal dependencies in manipulated voice signals [10].

2. Literature Review

Deepfake is a combination of *deep learning* and forgery (*fake*) that uses artificial intelligence algorithms to create manipulated media such as images, videos, and audio that resemble authentic content [11]. The early development of *deepfake* technology originated from facial simulation systems and later evolved into synthetic voice generation, making audio manipulation one of the fastest-growing threats in digital security [12].

Voice *deepfake* refers to synthetic speech generated by *machine learning* models that imitate real human voices. Detecting this manipulation requires effective feature extraction methods capable of revealing hidden signal patterns. One of the most commonly used representations is the *spectrogram*, which visualizes the relationship between time, frequency, and signal intensity in audio signals [13]. Through *Discrete Fourier Transform* (DFT), audio signals are transformed from the time domain into the frequency domain, allowing hidden manipulation patterns to be observed more clearly [14]. Another important feature extraction method is *Mel-Frequency Cepstral Coefficients* (MFCC), which represents speech based on human auditory perception. MFCC emphasizes frequency components that are more sensitive to human hearing and includes stages such as *pre-emphasis*, *framing*, *windowing*, FFT, *Mel filter bank*, and *Discrete Cosine Transform* (DCT) [15]. This method is widely used because it effectively captures important speech characteristics for classification tasks.

The proposed detection model is built upon *Artificial Neural Network* (ANN), a computational model inspired by the biological neural system of the human brain [16]. ANN consists of interconnected neurons that process information through weighted connections and activation functions. In modern classification tasks, *deep learning* extends ANN by using multiple hidden layers to automatically learn complex feature representations from data [17]. One of the most widely used architectures in audio classification is *Convolutional Neural Network* (CNN), which is highly effective for extracting spatial features from two-dimensional inputs such as spectrograms [18]. Meanwhile, *Recurrent Neural Network* (RNN) is designed to process sequential data by maintaining hidden states across time steps. To overcome the *vanishing gradient* problem in standard RNN, *Long Short-Term Memory* (LSTM) and *Bidirectional LSTM* (BiLSTM) are commonly used because they can preserve long-term dependencies and process information in both forward and backward directions [19].

The combination of CNN and RNN has shown strong performance in audio classification because CNN captures local spatial patterns while BiLSTM learns temporal dependencies. This hybrid architecture was introduced by Donahue et al. through the *Long-term Recurrent Convolutional Networks* (LRCN), which successfully combined CNN and RNN for sequential data analysis [20]. This architecture became the foundation for many later studies in audio classification and voice *deepfake* detection. Therefore, this study applies a CNN–RNN (BiLSTM) architecture integrated with MFCC and *spectrogram* features to improve voice *deepfake* detection, aiming to achieve higher accuracy and stronger generalization compared to conventional single-model approaches.

Several studies related to deepfake detection and speech signal processing have been conducted using deep learning approaches. Research conducted by Abidin et al. applied a combination of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) for deepfake detection. The study demonstrated that CNN is capable of extracting spatial characteristics, while LSTM can learn temporal changes from sequential data. However, the research focused on video-based deepfake detection rather than synthetic voice analysis, leaving opportunities for further exploration in audio deepfake detection [21]. In speech processing research, MFCC has been widely applied as an acoustic feature extraction method because it can represent important characteristics of human speech signals. Hermanto and Sen implemented MFCC combined with CNN for speech recognition tasks, showing that MFCC-based representations can provide effective input features for deep learning classification models. Nevertheless, the research focused on speech recognition problems and did not address the differences between natural and synthetic speech signals [22].

Research related to synthetic speech detection has also developed in Asian speech environments. Adila et al. investigated spoof voice detection using Indonesian and Thai speech data by extracting MFCC, LFCC, and CQCC features combined with machine learning classifiers. The study showed that language characteristics and speaker variations affect spoofing detection performance, indicating the importance of developing models that can learn more complex acoustic representations for diverse speech conditions [23]. Furthermore, deep learning-based approaches have increasingly been applied for detecting manipulated audio. Previous studies have demonstrated that spectrogram representations are effective because audio signals can be transformed into time-frequency representations that preserve frequency variations and artificial generation artifacts. CNN-based models are commonly used to extract these spatial patterns, while recurrent architectures such as LSTM and BiLSTM are utilized to capture temporal dependencies within speech sequences.

Although several studies have investigated deepfake detection using CNN, LSTM, and acoustic features, most existing approaches still focus on individual feature representations or separate spatial and temporal modeling. Research combining multiple acoustic features, such as spectrogram and MFCC, with a hybrid CNN–BiLSTM architecture for voice deepfake detection remains limited. Therefore, this research proposes a CNN–RNN (BiLSTM) model integrated with spectrogram and MFCC features to capture both frequency-based characteristics and temporal patterns of synthetic speech. This combination is expected to improve detection performance and provide better generalization for identifying AI-generated voices.

3. Methodology

3.1. Research Procedure

This study was conducted through several systematic stages to develop a deepfake voice detection system using a hybrid CNN–RNN (BiLSTM) model. The first stage involved designing the overall system architecture, including the workflow for processing original and deepfake voice data. The flowchart is presented in Fig. 1.

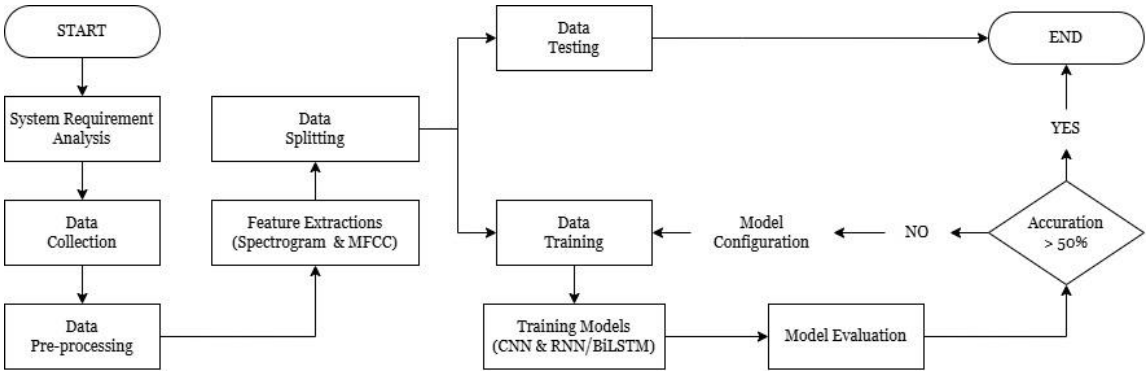

Fig. 1: Flowchart of the Research Procedure

Original voice data were collected through informal interviews and free conversations with participants, then processed to generate deepfake voice samples using artificial intelligence-based voice generation models. After data collection, preprocessing was performed through noise reduction, volume normalization, segmentation, format conversion, and sampling rate adjustment to ensure consistency and improve audio quality. The processed audio was then transformed into numerical representations using feature extraction techniques, namely *Mel-Frequency Cepstral Coefficients* (MFCC) and *spectrogram*, to capture both temporal and frequency characteristics of speech signals.

The extracted features were divided into training and testing sets using several split ratios to determine the most optimal model performance. The next stage involved designing the CNN–RNN (BiLSTM) architecture, followed by model training using multiple epochs and batch processing. Finally, model performance was evaluated using classification metrics such as confusion matrix, accuracy, precision, recall, F1-score, and ROC-AUC to determine the effectiveness of the proposed method.

3.2. Data Collection

The data used in this study were collected as primary data from human speech recordings. The objects of data collection were the voices of Indonesian speakers participating in informal interview sessions. The speech data were obtained through casual conversations, question-and-answer sessions, and spontaneous discussions to capture natural speaking characteristics. A total of 30 participants were involved in the data collection process, consisting of 15 male and 15 female speakers.

Each participant recorded at least 10 minutes of speech using high-quality audio recording settings with a bit rate of 320 kbps to preserve detailed acoustic information. The collected recordings represent authentic human speech containing variations in gender, accent, speaking style, emotional expression, and environmental conditions. The original human voice recordings were subsequently used as source data to generate deepfake voice samples using AI-based voice synthesis models from reliable sources. These generated samples represent artificial speech produced through voice manipulation techniques. Therefore, the dataset consisted of two data objects: (1) original human speech recordings and (2) AI-generated deepfake speech recordings derived from the original voice samples.

Each audio sample was labeled into two classes, namely original voice and deepfake voice, to support supervised learning. The recordings were segmented into 30-second audio samples, producing approximately 5 hours of original voice data and 5 hours of deepfake voice data, with a total dataset duration of approximately 10 hours. The inclusion of diverse speech characteristics was intended to improve the robustness and generalization capability of the proposed CNN–RNN (BiLSTM) detection model.

3.3. Data Pre-processing

Data preprocessing is an essential stage to improve data quality before feature extraction and model training. The first step was noise reduction to remove unnecessary background interference so that the model could focus on the main speech signal. Next, volume normalization was applied to ensure all recordings had consistent loudness levels and to reduce bias caused by recording intensity differences. Audio segmentation was then performed to divide long recordings into shorter and more relevant segments for analysis. All audio files were converted into a consistent format such as WAV to ensure compatibility with the processing pipeline. Sampling rates were also standardized to maintain uniformity across all data sources. Finally, data verification was carried out to ensure that the processed audio contained no corruption, missing labels, or preprocessing errors before entering the feature extraction stage.

3.4. Feature Extraction

Feature extraction was performed to convert audio signals into numerical representations that could be processed by the deep learning model. This study used two main feature extraction methods: *spectrogram* and *Mel-Frequency Cepstral Coefficients* (MFCC). Spectrograms were used to visualize frequency intensity over time and help identify hidden manipulation patterns in audio signals, while MFCC was used to capture perceptually important speech characteristics based on human hearing mechanisms. The combination of these two methods produced richer feature representations by preserving both spectral and temporal information. After extraction, feature normalization was applied to ensure all values were within similar scales and to prevent feature dominance during training. The final extracted features were stored in structured formats such as NumPy arrays and CSV files for efficient use during model training.

3.5. Data Training

After preprocessing and feature extraction, the next stage was model training using the hybrid CNN–RNN (BiLSTM) architecture. CNN was responsible for extracting spatial patterns from spectrogram inputs, particularly frequency distribution and local signal characteristics that distinguish original and manipulated voices. The output from CNN was then passed to the BiLSTM layer, which learned temporal dependencies such as intonation, rhythm, and long-term sequential speech patterns. The model architecture started with the input layer, followed by Conv2D layers, pooling layers, BiLSTM layers, dense layers, and finally an output layer using sigmoid or softmax activation for binary classification. Model compilation included the selection of loss functions, optimizers, and evaluation metrics. Data validation was performed using several split ratios, namely 60:40, 70:30, and 80:20, to identify the best-performing configuration. Training was conducted using multiple epochs and mini-batches until convergence was achieved and validation performance stabilized.

3.6. Model Evaluation

After the training process was completed, the model was evaluated using testing data to measure its effectiveness in detecting deepfake voice. Several evaluation metrics were used, including confusion matrix, accuracy, precision, recall, F1-score, and ROC-AUC. The confusion matrix compares prediction results with actual labels and consists of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).

		Predicted	
		Real (0)	Deepfake (1)
Actual	Real (0)	TP	FP
	Deepfake (1)	FN	TN

Fig. 2: Confusion Matrix Structure

Accuracy measures the overall correctness of classification results. Precision evaluates how accurately the model predicts deepfake samples as positive cases, while recall measures the model's ability to detect all actual deepfake samples. F1-score provides a balanced measurement between precision and recall. ROC-AUC was used to evaluate the model's capability in distinguishing between original and deepfake classes, where values closer to 1 indicate stronger classification performance.

4. Results and Discussion

4.1. System Requirement Analysis

This study was designed to develop a deepfake voice detection system capable of distinguishing between original human speech (real) and AI-generated synthetic speech (deepfake). The entire experimental workflow was arranged systematically, starting from hardware preparation, software configuration, dataset collection, preprocessing, feature extraction, model training, and final system implementation. The research utilized medium-level hardware specifications to ensure efficient feature extraction and model training performance. The main computing device used was an Acer Aspire 3 A314-22 laptop equipped with an AMD Athlon Silver 3050U processor, 8 GB DDR4 RAM, 256 GB SSD, and integrated AMD Radeon Graphics running on Windows 11 Home 64-bit. Audio recording activities were supported using a Xiaomi Redmi 10C smartphone and a Belcore D80 Magnetic Wireless Dual Microphone to ensure clear voice acquisition during the interview process. The software environment consisted of Audacity for audio recording, trimming, and cleaning; Python 3.10 with Visual Studio Code for preprocessing and feature engineering; and Google Colab with cloud GPU acceleration for deep learning model training. Several Python libraries were integrated to support the full pipeline of the research process.

Table 1: Main Python Libraries Used

Library	Main Function
Librosa	Audio feature extraction (MFCC and Spectrogram)
NumPy	Numerical computation and array operations
Pandas	Data manipulation and tabular analysis
Matplotlib	Data visualization and spectrogram plotting
TensorFlow	CNN–RNN (BiLSTM) model development
Scikit-learn	Model evaluation and classification metrics

4.2. Dataset Collection and Audio Standardization

The dataset consisted of 60 original audio files, including 30 real voice recordings collected from interviews with university students and 30 deepfake voice samples generated using Minimax.io AI Voice Generator through voice cloning and Text-to-Speech (TTS). All audio files were standardized into WAV format to preserve acoustic integrity and ensure consistent feature extraction.

Table 2: Audio Standardization Specifications

Parameter	Value
Format	WAV
Sampling Rate	44,100 Hz
Channel	Mono
Bit Depth	16-bit PCM
Average Duration	300 seconds

From format standardization, segmentation was performed to divide each 300-second recording into fixed-length segments of 30 seconds. The Result of sample from segmentation audio in Fig 3.

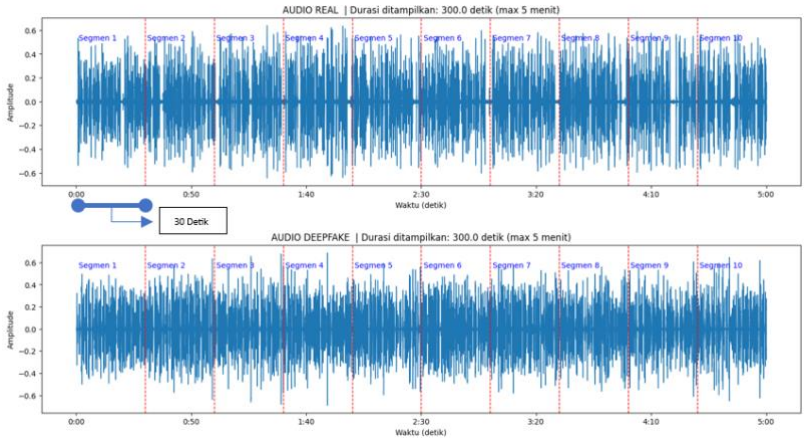

Fig. 3: Audio Segmentation Result (30 Seconds per Segment)

After segmentation using 30-second intervals, each audio file produced 10 segments, resulting in 600 audio segments before augmentation.

4.3. Data Augmentation

To improve model generalization and reduce overfitting, data augmentation was applied to the 600 segmented audio files. Augmentation increases dataset diversity by creating new training samples without changing the semantic content of the speech signal. Five augmentation techniques were implemented using Librosa, NumPy, and SoundFile.

Table 3: Audio Data Augmentation Techniques

Augmentation Technique	Parameter Used	Purpose
Time Shifting	0.1 – 0.5 seconds	Simulate speaking delay variations
Pitch Shifting	−2 to +2 semitones	Simulate pitch differences
Time Stretching	0.9 – 1.1 rate	Simulate speaking speed variations
Additive Gaussian Noise	0.005 amplitude factor	Simulate environmental noise
Amplitude Scaling	0.8 – 1.2 scale	Simulate recording volume differences

Each of the 600 segmented audio files was augmented using all five techniques. Thus, the augmentation process produced 3000 new audio samples. This significant increase in data diversity helped improve model robustness when detecting both real and synthetic voice patterns.

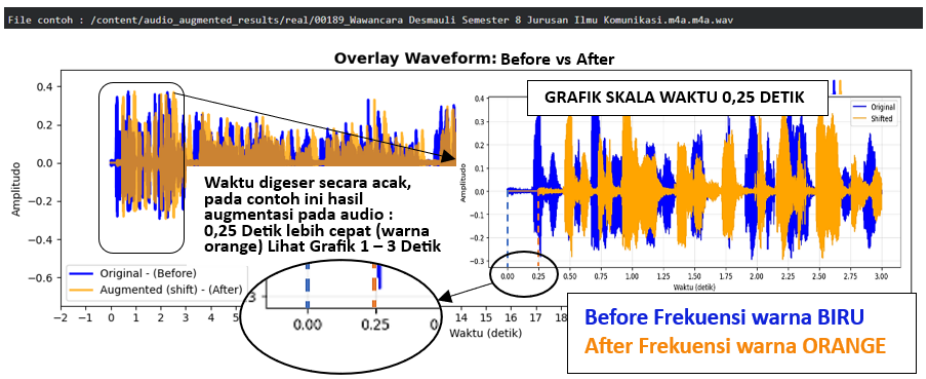

Fig. 4: Waveform Representation of Audio After Time Shifting Augmentation

Time shifting is an augmentation technique that modifies the temporal position of an audio signal by shifting the waveform along the time axis. In this study, the audio signal was randomly shifted within a range of 0.1 to 0.5 seconds. This process simulates variations in the starting position of speech recordings while preserving the original speech characteristics. Figure 4 shows an example of the waveform transformation after applying time shifting. The waveform pattern remains similar to the original signal; however, its position changes along the time axis.

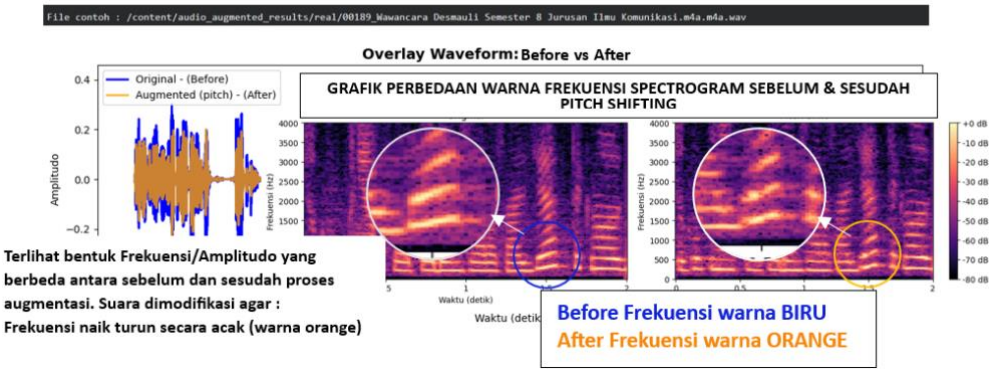

Fig. 5: Waveform Representation of Audio After Pitch Shifting Augmentation

Pitch shifting modifies the fundamental frequency of an audio signal by increasing or decreasing the pitch level without changing the speech duration. In this research, the pitch value was randomly adjusted between −2 and +2 semitones. This augmentation represents natural variations in human speech, such as differences in speaker intonation and vocal characteristics. Figure 5 illustrates the waveform representation after pitch shifting is applied.

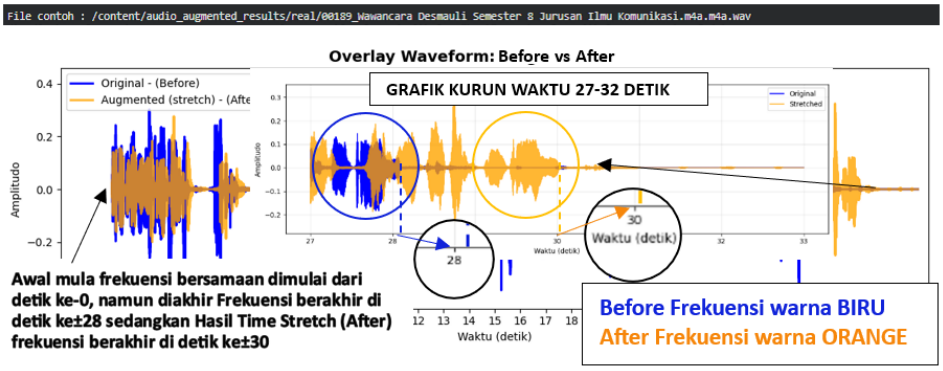

Fig. 6: Waveform Representation of Audio After Time Stretching Augmentation

Time stretching changes the duration of an audio signal by modifying its playback speed while maintaining the original pitch. The stretching rate was randomly selected between 0.9 and 1.1, where values below 1 slow down the audio and values above 1 accelerate the audio approximately by 10% from the original duration. Figure 6 presents the waveform variation after applying time stretching augmentation. The main difference can be observed in the duration and distribution of the signal over time.

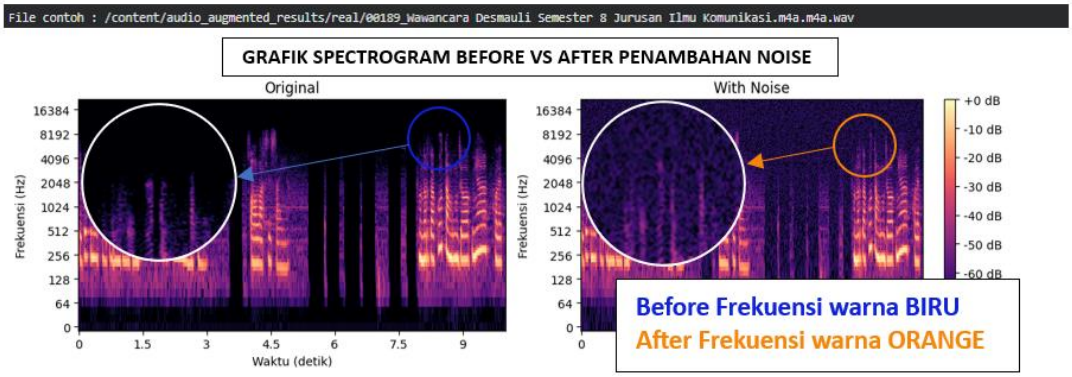

Fig. 7: Waveform Representation of Audio After Additive Gaussian Noise Augmentation

Additive Gaussian Noise was applied by adding random noise components to the original audio signal. The added noise has a relatively small amplitude, making it difficult to observe directly from the waveform representation. However, the differences become more visible when the signal is converted into a spectrogram representation because noise introduces additional energy distribution across frequency components. Figure 7 shows the waveform representation after Gaussian noise augmentation.

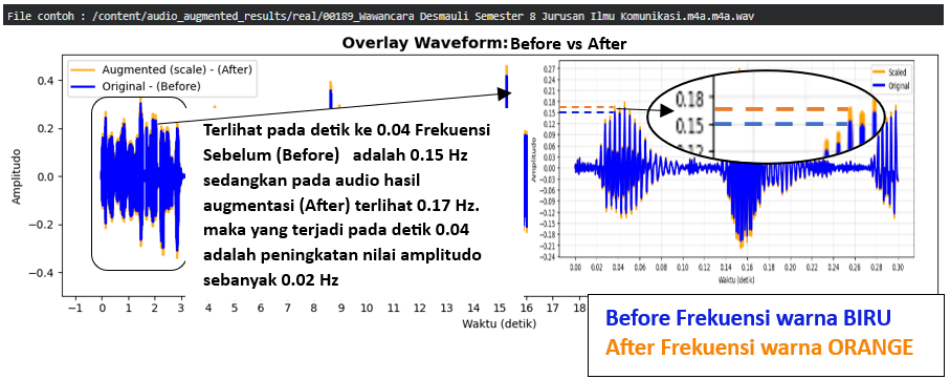

Fig. 8: Waveform Representation of Audio After Amplitude Scaling Augmentation

Amplitude scaling modifies the volume level of an audio signal by multiplying the waveform amplitude with a scaling factor. In this study, the scaling factor was randomly selected between 0.8 and 1.2, resulting in volume variations of approximately ±20% from the original amplitude. This augmentation simulates differences in recording distance, microphone sensitivity, and environmental recording conditions. Figure 8 shows the waveform changes after amplitude scaling.

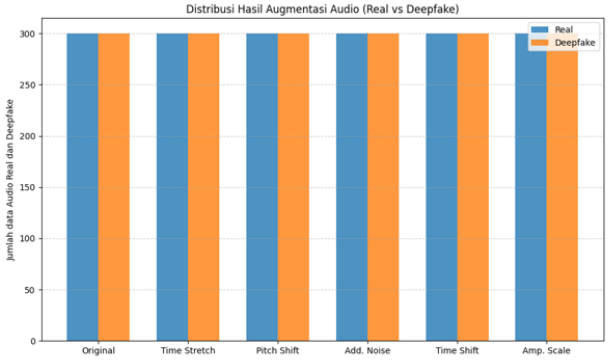

Fig. 9: Augmentation Result and Final Audio Distribution

4.4. Result of Feature Extraction

After augmentation, all 3600 audio files were transformed into numerical representations using MFCC (Mel-Frequency Cepstral Coefficients) and Log-Mel Spectrogram. MFCC captured speech timbre and cepstral information, while Log-Mel Spectrogram represented temporal-frequency energy distribution. Their combination improved deepfake pattern recognition significantly.

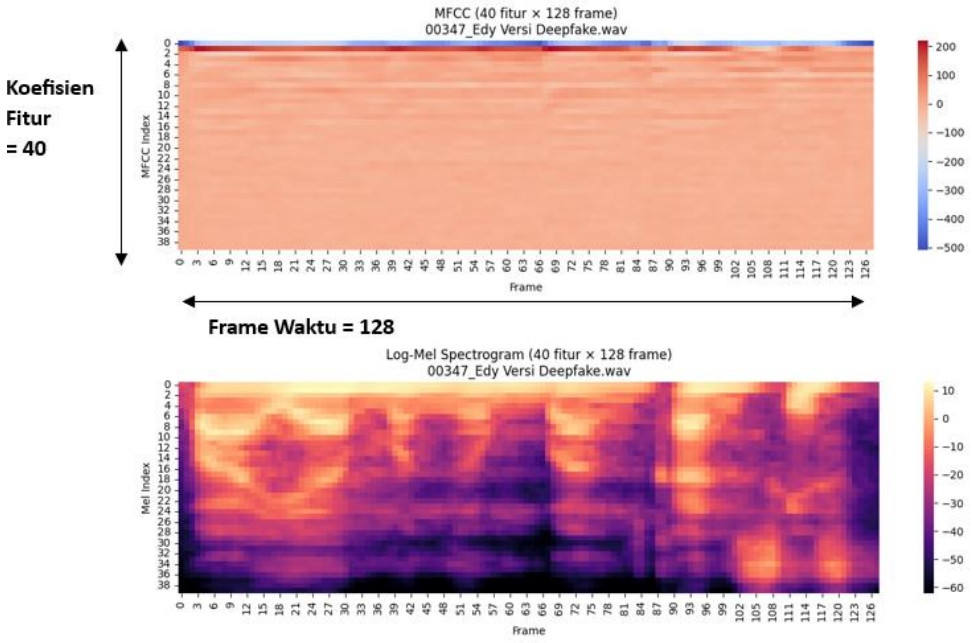

Fig. 10: Feature Extraction Result

4.5. Data Splitting

Before training the classification model, all extracted features were normalized to ensure balanced feature contribution during the learning process. Without normalization, features with large numerical ranges may dominate smaller-scale features and reduce model stability. StandardScaler was used to normalize all features using Z-score normalization. Since the dataset was already balanced between real and deepfake classes, no additional class balancing technique was required. To evaluate the influence of training data proportion on model performance, three train-test ratio scenarios were applied 60:40, 70:30 and 80:20. These scenarios allowed objective comparison of model stability and generalization performance.

Table 4: Dataset Splitting Results

Data Ratio	Training Data	Testing Data	Train Shape	Test Shape
60:40	2160	1440	(2160, 128, 80)	(1440, 128, 80)
70:30	2520	1080	(2520, 128, 80)	(1080, 128, 80)
80:20	2880	720	(2880, 128, 80)	(720, 128, 80)

The 80:20 ratio provided the highest amount of training data while maintaining sufficient testing samples for reliable evaluation. This stage became the foundation for model training using CNN–BiLSTM, where the impact of training data proportion was analyzed comprehensively in the next section.

4.6. CNN–BiLSTM Integration

The hybrid CNN–BiLSTM architecture integrated the strengths of both models. CNN for spatial-frequency feature extraction, BiLSTM for temporal sequence learning. This integration allowed the system to analyze voice patterns more comprehensively.

Table 5: CNN–BiLSTM Integration Flow

Stage	Process	Description
1	Input Data	Audio features with shape (128, 80, 1)
2	Spatial Extraction (CNN)	CNN learns frequency-energy patterns
3	Reshape	CNN output converted into sequential form
4	Temporal Analysis (BiLSTM)	BiLSTM learns temporal dependencies
5	Dense + Output	Final classification into REAL or FAKE

This hybrid architecture was highly effective because deepfake detection requires both spatial understanding of frequency structure and temporal understanding of speech evolution over time. Using CNN alone would miss sequential dependencies, while using BiLSTM alone would weaken local feature extraction. Thus, CNN–BiLSTM provided significantly better generalization and classification performance.

4.7. Model Evaluation and Result

The model evaluation stage aimed to measure the ability of the proposed CNN–BiLSTM architecture in distinguishing between real voices and deepfake voices based on previously extracted MFCC and Log-Mel Spectrogram features. Evaluation was performed using unseen testing data to ensure that the results reflected the model's true generalization capability rather than memorization of training samples. The testing dataset originated from the complete preprocessing pipeline, including audio conversion, segmentation, augmentation, normalization, and feature extraction.

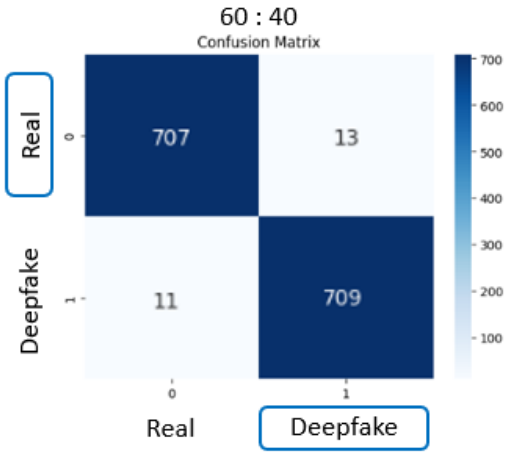

Fig. 11: Confusion Matrix Results Using 60:40 Training and Testing Data Distribution

Based on the evaluation results, the model correctly classified 707 real voice samples and 709 deepfake voice samples. Meanwhile, 13 real voice samples were incorrectly classified as deepfake, and 11 deepfake voice samples were misclassified as real voice.

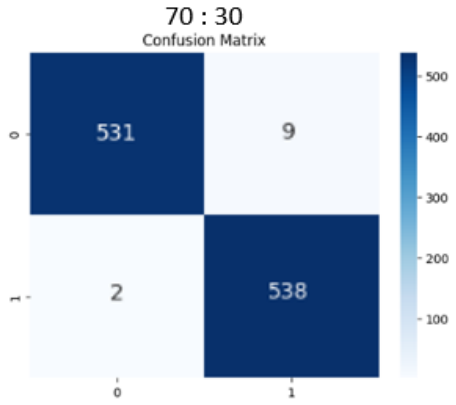

Fig. 12: Confusion Matrix Results Using 70:30 Training and Testing Data Distribution

Figure 12 shows the confusion matrix evaluation using a 70:30 training and testing data split. The model correctly identified 531 real voice samples and 538 deepfake voice samples. The number of incorrect predictions consisted of 9 real voice samples classified as deepfake voice and only 2 deepfake voice samples classified as real voice. These results demonstrate that increasing the proportion of training data improves the model's ability to recognize deepfake patterns.

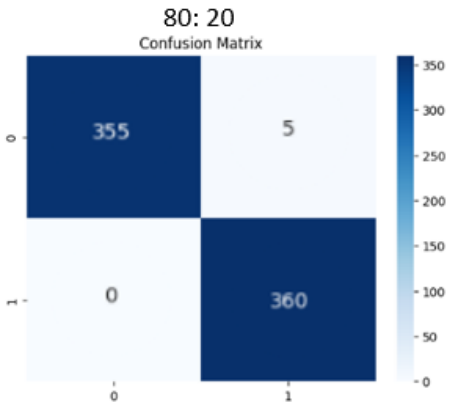

Fig. 13: *C*onfusion Matrix Results Using *80:20* Training and Testing Data Distribution

Confusion Matrix was used to evaluate prediction correctness for both classes (real and fake) and to analyze false positive and false negative rates. Among the three ratio scenarios, the 80:20 ratio produced the best classification performance. The model successfully classified 355 real voice samples correctly with only 5 misclassifications and 360 fake voice samples correctly with 0 misclassifications.

The comparison among all ratios showed that increasing training data proportion improved prediction stability and reduced classification errors significantly. To further evaluate classification quality, Precision, Recall, and F1-Score were calculated for each class under all ratio scenarios.

Table 6: Classification Report for All Ratios

Ratio	Class	Precision	Recall	F1-Score	Support
60:40	REAL	0.98	0.98	0.98	720
	FAKE	0.98	0.98	0.98	720
	Total Accuracy	0.983	—	—	1440
70:30	REAL	1.00	0.98	0.99	540
	FAKE	0.98	1.00	0.99	540
	Total Accuracy	0.989	—	—	1080
80:20	REAL	1.00	0.99	0.99	360
	FAKE	0.99	1.00	0.99	360
	Total Accuracy	0.993	—	—	720

The results clearly show that the 80:20 ratio achieved the highest overall accuracy 99.3% followed by 70:30 in 98.9% and 60:40 in 98.3%. To evaluate training stability and possible overfitting, training accuracy, validation accuracy, training loss, and validation loss were analyzed for all three ratios across 50 epochs.

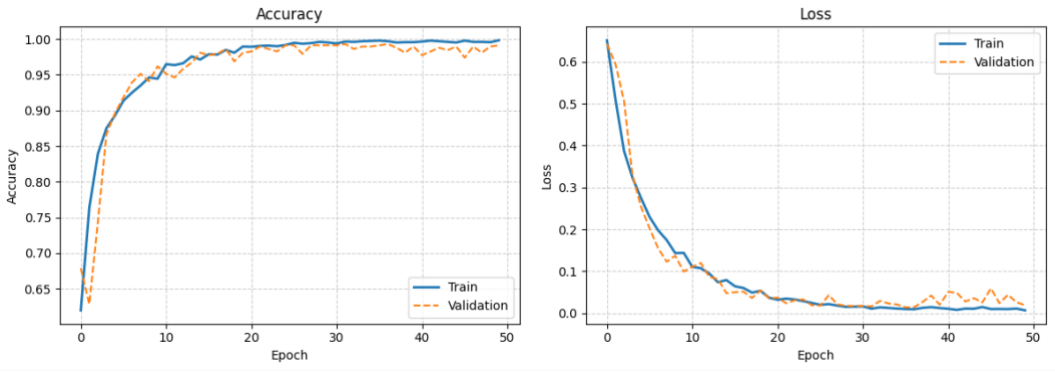

Fig. 14: Sample of Accuracy Graph in the 80:20 Train-Test Ratio Model Evaluation

Training and validation curves were nearly identical, indicating near-perfect convergence and excellent model stability. Although smaller validation size may introduce slight evaluation bias, the overall performance strongly confirmed model effectiveness.

4.8. Model Testing Using Gradio

The model testing stage was conducted to evaluate the system's ability to classify original (real) and manipulated (deepfake) voice recordings. Testing was implemented through an interactive interface using Gradio, allowing users to upload audio files, perform automatic feature extraction, and obtain prediction results visually and efficiently. The Gradio interface was designed to simplify user interaction with the deepfake voice detection system without requiring manual programming commands. As shown in Figure 4.21, the interface consists of six main components: (1) audio file upload, (2) uploaded file display, (3) submit button, (4) clear button, (5) prediction and probability output, and (6) feature visualization panels for Mel-Spectrogram and MFCC.

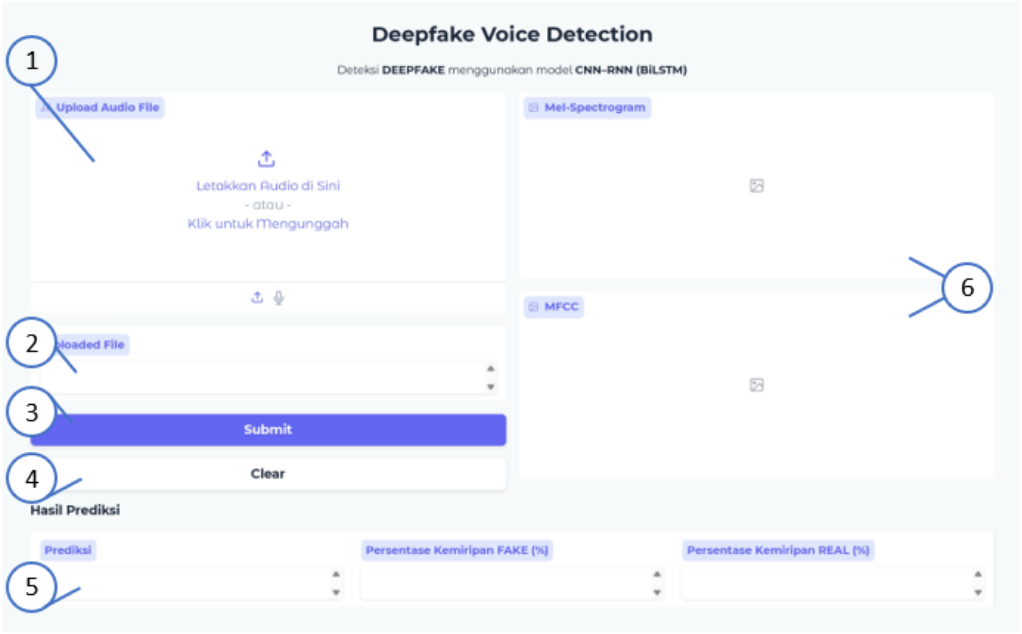

Fig. 15: Interface Gradio of Detection Audio Deepfake System

The upload component supports several audio formats, including *.wav*, *.mp3*, and *.flac*. After the file is uploaded, the system displays the filename as confirmation that the input has been successfully received. By pressing the submit button, the system performs preprocessing steps, including normalization, feature extraction, and inference using the CNN–BiLSTM model. The prediction result is then displayed as either Real or Fake, along with the confidence percentage for each class.

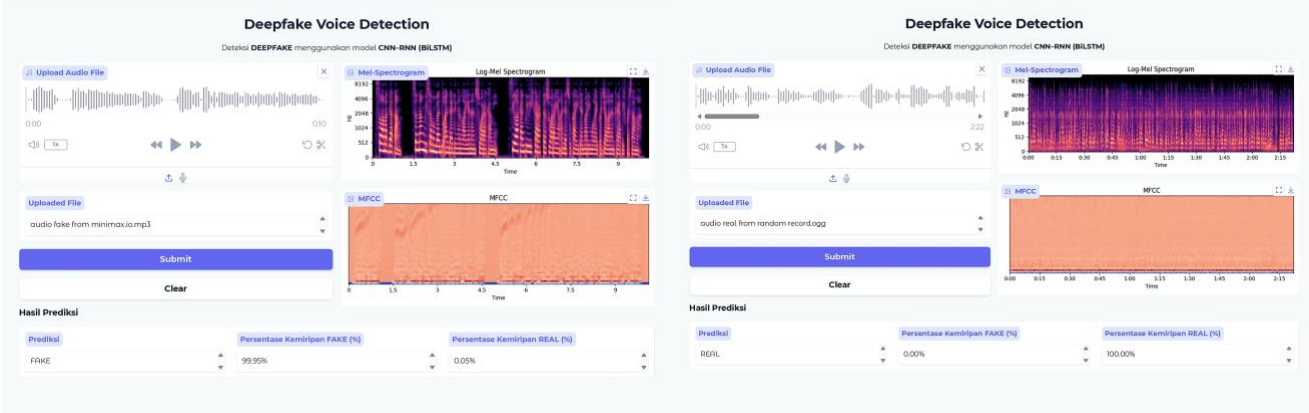

Fig. 4: Audio Testing Result in the Detection System

The inference result showed a prediction of Fake with a fake probability of 99.95%, indicating that the model successfully identified the synthetic speech patterns commonly found in deepfake audio with very high confidence. The system produced a prediction of Real with a fake probability of 0%, indicating that the model correctly recognized the input as authentic human speech. This result confirms the effectiveness of the proposed CNN–BiLSTM model in distinguishing natural voice signals from AI-generated synthetic voices.
However, several additional experiments using Kaggle audio samples labeled as fake, including voice-cloned and voice-changed recordings, were still classified as Real with approximately 20% fake probability. This indicates that although the model achieved high overall accuracy, further improvement is still required, particularly by increasing dataset diversity and including more variations of deepfake audio generation methods to enhance model generalization and robustness.

5. Conclusion

This study successfully developed a deepfake voice detection model using a hybrid CNN–BiLSTM architecture. The model effectively distinguishes real and fake audio by leveraging Mel-Spectrogram and MFCC features, where CNN captures time–frequency patterns and BiLSTM models temporal dependencies. The combination of these features improves classification performance in identifying speech authenticity.

Experimental results show that an 80:20 train–test split yields the best performance, achieving 99.3% accuracy with an AUC close to 1.0, indicating high reliability in detecting deepfake audio. However, limitations were observed in generalization, particularly when tested on external datasets with unseen deepfake variations such as voice cloning and voice conversion. Therefore, future improvements should focus on expanding dataset diversity, evaluating with external data, and developing real-time detection capabilities to enhance robustness and practical applicability.

References

[1] S. Widya Sasana, "Etika Penggunaan Teknologi AI Menurut Paul Ricoeur sebagai Realisasi Hidup Baik," *Paradigma*, vol. 30, no. 1, pp. 17–27, 2023. doi: 10.33503/paradigma.v30i1.4020.
[2] R. Angelika Septi Rahayu and H. Santoso, "Analisis Gambar Wajah Palsu: Mendeteksi Keaslian Gambar yang Dimanipulasi Menggunakan Metode Variasional Autoencoder dan Forensik Jaringan Neural Deep," *Sibatik Journal*, vol. 2, no. 9, 2023. doi: 10.54443/sibatik.v2i9.1312.
[3] Z. Khanjani, G. Watson, and V. P. Janeja, "Audio Deepfakes: A Survey," *Frontiers in Big Data*, vol. 5, p. 1001063, 2023. doi: 10.3389/fdata.2022.1001063.
[4] C. Stupp, "Fraudsters Used AI to Mimic CEO's Voice in Unusual Cybercrime Case," *WSJ Pro Cyber Security*, Aug. 30, 2019.
[5] Kementerian Komunikasi dan Informatika, "Antisipasi Deep Fake, Wamen Nezar Patria: Kominfo Lindungi Kelompok Rentan," 2023. [Online]. Available: https://www.kominfo.go.id/
[6] K. T. Mai, S. Bray, T. Davies, and L. D. Griffin, "Warning: Humans Cannot Reliably Detect Speech Deepfakes," *PLOS ONE*, vol. 18, no. 8, p. e0285333, 2023. doi: 10.1371/journal.pone.0285333.
[7] B. Dolhansky, J. Bitton, B. Pflaum, J. Lu, R. Howes, M. Wang, and C. C. Ferrer, "The DeepFake Detection Challenge (DFDC) Dataset," 2020. [Online]. Available: http://arxiv.org/abs/2006.07397
[8] B. Malolan, A. Parekh, and F. Kazi, "Explainable Deep-Fake Detection Using Visual Interpretability Methods," in *Proc. 3rd Int. Conf. Information and Computer Technologies (ICICT)*, 2020, pp. 289–293. doi: 10.1109/ICICT50521.2020.00051.
[9] S. Y. Lim, D. K. Chae, and S. C. Lee, "Detecting Deepfake Voice Using Explainable Deep Learning Techniques," *Applied Sciences*, vol. 12, no. 8, 2022. doi: 10.3390/app12083926.
[10] M. U. Tanveer, K. Munir, M. Amjad, A. U. Rehman, and A. Bermak, "Unmasking the Fake: Machine Learning Approach for Deepfake Voice Detection," *IEEE Access*, 2024. doi: 10.1109/ACCESS.2024.3521026.
[11] L. Gaur, *DeepFakes*. Boca Raton: CRC Press, 2022. doi: 10.1201/9781003231493.
[12] S. Karnouskos, "Artificial Intelligence in Digital Media: The Era of Deepfakes," *IEEE Transactions on Technology and Society*, vol. 1, no. 3, pp. 138–147, 2020. doi: 10.1109/TTS.2020.3001312.
[13] S. Y. Hartono, S. Suyuti, Bambang, P. Asmara, A. Ashad, and N. K. Hamzidah, "Analisis Spektogram Sinyal Suara Asli dan Suara Hasil Konversi Berbasis Derivative Gelombang Glotal," *Jurnal Teknologi Elekterika*, vol. 20, no. 2, pp. 24–28, 2023.
[14] P. Stoica and R. Moses, *Spectral Analysis of Signals*. Upper Saddle River, NJ: Pearson Prentice Hall, 2005.
[15] S. Ali, S. Tanweer, S. Khalid, and N. Rao, "Mel Frequency Cepstral Coefficient: A Review," Mar. 17, 2021. doi: 10.4108/eai.27-2-2020.2303173.
[16] E. Grossi and M. Buscema, "Introduction to Artificial Neural Networks," *European Journal of Gastroenterology and Hepatology*, vol. 19, no. 12, pp. 1046–1054, 2007. doi: 10.1097/MEG.0b013e3282f198a0.

[17] A. Raup, W. Ridwan, Y. Khoeriyah, Q. Yuliati Zaqiah, and U. Islam Negeri Sunan Gunung Djati Bandung, "Deep Learning dan Penerapannya dalam Pembelajaran," 2022.
[18] S. Ilahiyah and A. Nilogiri, "Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network," 2018.
[19] R. Onsu, D. Febrian, and F. Diane, "Implementasi Bi-LSTM Dengan Ekstraksi Fitur Word2vec Untuk Pengembangan Analisis Sentimen Aplikasi Identitas Kependudukan Digital," *Jurnal Teknologi Terpadu*, vol. 10, no. 1, pp. 46–55, 2024.
[20] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-term Recurrent Convolutional Networks for Visual Recognition and Description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 677–691, 2014. doi: 10.1109/TPAMI.2016.2599174.
[21] M. I. Abidin, I. Nurtanio, and A. Achmad, "Deepfake Detection in Videos Using Long Short-Term Memory and CNN ResNext," ILKOM Jurnal Ilmiah, vol. 14, no. 3, pp. 285–292, 2022. doi: 10.33096/ilkom.v14i3.1254.
[22] H. Hermanto and T. W. Sen, "Syllable-Based Javanese Speech Recognition Using MFCC and CNNs: Noise Impact Evaluation," Jurnal Teknik Informatika, vol. 18, no. 1, 2025. doi: 10.15408/jti.v18i1.41067.
[23] A. Adila, C. O. Mawalim, and M. Unoki, "Detecting Spoof Voices in Asian Non-Native Speech: An Indonesian and Thai Case Study," arXiv preprint arXiv:2412.01040, 2024.