VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANASANGAMA, BELAGAVI - 590018

**Project Work Phase II Report (21AML183)**
**On**

# TRUEE TONE: AUDIO AUTHENTICITY DETECTION

*Submitted in partial fulfillment for the award of degree of*

**Bachelor of Engineering**

**in**

**ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING**

Submitted by

Deepak K L [1BG21AI031]
Dhruva Kashyap [1BG21AI032]

**Under the guidance of**

**Mr. Mohanesh B M**

Assistant Professor, Dept. of AIML, BNMIT, Bengaluru

Vidyayāmruthamashnuthe

B.N.M. Institute of Technology

**An Autonomous Institution under VTU**
Approved by AICTE, Accredited as grade A Institution by NAAC. All eligible branches – CSE, ECE, EEE, ISE & Mech. Engg. are Accredited by NBA for academic years 2018-19 to 2024-25 & valid upto 30.06.2025. URL: www.bnmit.org

**Department of Artificial Intelligence and Machine Learning**

**2024 – 2025**

B.N.M. Institute of Technology

**An Autonomous Institution under VTU**

Approved by AICTE, Accredited as grade A Institution by NAAC. All eligible branches – CSE, ECE, EEE, ISE & Mech. Engg. are Accredited by NBA for academic years 2018-19 to 2024-25 & valid upto 30.06.2025. URL: www.bnmit.org

**Department of Artificial Intelligence and Machine Learning**

Vidyayāmruthamashnuthe

# CERTIFICATE

This is to certify that the project work phase – II titled **TrueeTone: Audio Authenticity Detection** carried out by **Dhruva Kashyap (1BG21AI032),** the bonafide students of VIII Semester B.E., **B.N.M Institute of Technology,** in partial fulfillment for the award of degree in **Bachelor of Engineering** in ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING of the **Visvesvaraya Technological University**, Belagavi during the year 2024-25. This report has been approved as it satisfies the academic requirements in respect of Project Work Phase – II (21AML183) prescribed for the said degree.

| | | |
|---|---|---|
| **Mr. Mohanesh B M** | **Dr. Sheba Selvam** | **Dr. S.Y. Kulkarni** |
| Assistant Professor | Professor and HOD | Additional Director |
| Dept. of AIML | Dept. of AIML | and Principal |
| BNMIT, Bengaluru | BNMIT, Bengaluru | BNMIT, Bengaluru |

| | **Name** | **Signature** |
|---|---|---|
| **Examiner 1:** | | |
| **Examiner 2:** | | |

# DECLARATION

I, Dhruva Kashyap (1BG21AI032), hereby declare that the Project Work Phase-II (21AML183) entitled "TrueeTone: Audio Authenticity Detection" has been independently carried out by me under the guidance of Mr. Mohanesh B M, Assistant Professor, Department of AIML, BNM Institute of Technology, Bengaluru, in partial fulfillment of the requirements of the degree of Bachelor of Engineering in Artificial Intelligence & Machine Learning of the Visvesvaraya Technological University, Belagavi.

I further declare that I have not submitted this report either in part or in full to any other university for the reward of any degree.

**Place: Bengaluru**
**Date:**

Dhruva Kashyap

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success.

I would like to thank **Shri Narayan Rao R. Maanay,** Secretary, BNMEI, Bengaluru for providing an excellent academic environment in the College.

I would like to sincerely thank **Prof. T. J. Rama Murthy,** Director, BNMIT, Bengaluru, for having extended his support and encouraging me during the course of the work.

I would like to sincerely thank **Dr. S.Y. Kulkarni,** Additional Director and Principal BNMIT, Bengaluru, for having extended his support and encouraging my during the course of the work.

I would like to express my gratitude to **Prof. Eishwar N. Maanay,** Dean, BNMIT, Bengaluru for his relentless support, guidance and assistance.

I would like to thank **Dr. Krishnamurthy G.N,** Deputy Director, BNMIT, Bengaluru for his constant encouragement.

I would like to thank **Dr. Sheba Selvam,** Professor and Head of the Department of Artificial Intelligence and Machine Learning, BNMIT, Bengaluru who has shared her opinions and thoughts which helped me in completion of my project successfully.

I would also like to thank my Guide **Mr. Mohanesh B M,** Assistant Professor, Department of Artificial Intelligence and Machine Learning, BNMIT, Bengaluru for guiding in asystematic manner.

Finally, I would like to thank all technical and non-technical faculty members of Department of Artificial Intelligence and Machine Learning, BNMIT, Bengaluru, for their support. I would like to thank my Family and Friends for their unfailing moral support and encouragement.

**DHRUVA KASHYAP**
**1BG21AI032**

# ABSTRACT

The rapid evolution of artificial intelligence and deep learning technologies has revolutionized various domains, including speech synthesis and voice generation. While these advancements have brought significant benefits, they have also introduced new challenges—most notably, the creation of AI-generated fake audio, commonly known as audio deepfakes. These synthetically produced voices are nearly indistinguishable from real human speech, posing serious threats to digital security, personal identity, and public trust. In response to this growing concern, we developed TrueeTone, an AI-powered solution that can accurately determine whether an audio sample is Real (Human) or Fake (AI-generated). The system is built using a multi-model architecture, integrating three powerful classification techniques. The first is a Dense Neural Network (DNN) trained using MFCC (Mel Frequency Cepstral Coefficients) to learn low-level spectral patterns in audio. The second is a Convolutional Neural Network (CNN), also leveraging MFCC features, but focusing on spatial feature extraction through mel-spectrograms. The third model is the pre-trained Wav2Vec 2.0 (wav2vec-960h-base), which extracts high-level contextual representations directly from raw waveforms. This diverse model ensemble enables TrueeTone to achieve high precision and robustness across various audio types. The frontend, developed using Streamlit and deployed via Ngrok on Google Colab, provides a clean and intuitive interface where users can upload or record audio. The system then visualizes the mel-spectrogram, processes the input through all three models, and displays individual predictions as well as the final ensemble result using majority voting logic. A detailed report is also generated summarizing model outputs, giving users transparency and interpretability. Comprehensive testing methodologies—including whitebox, blackbox, unit testing, and integration testing—were applied to ensure system reliability, functional correctness, and seamless model integration. The Wav2Vec model emerged as the most accurate, followed by CNN and DNN, with the final ensemble accuracy reaching up to 96%. TrueeTone is not only a technical innovation but also a socially impactful tool, contributing to media forensics, cybersecurity, and digital ethics. It aligns with Sustainable Development Goals like Industry, Innovation, and Infrastructure and Peace, Justice, and Strong Institutions by ensuring responsible and secure use of AI technologies. This system lays the groundwork for future expansions, including multi-language detection, real-time voice verification, and video deepfake integration. In conclusion, TrueeTone provides a scalable, reliable, and innovative solution to address the growing challenge of AI-generated audio content in today's digital world.

# Table of Contents

# List of Figures

# List of Tables

# CHAPTER 1
# INTRODUCTION

# CHAPTER 1

# INTRODUCTION

## 1.1 Motivation

The accelerated development of artificial intelligence (AI) has transformed different fields, such as speech synthesis and sound generation. Following the advent of deep learning methods, artificial intelligence-generated sounds, commonly known as deepfake audio, have become indistinguishable and even more realistic compared to human voice. Although this technology is used in hundreds of beneficial applications across assistive technologies, entertainment, and speech synthesis, it also presents critical threats. Deepfake audio can be misused for malicious activities such as misinformation, impersonation, and fraudulent schemes, raising significant ethical and security concerns. The motivation behind this research is the urgent need to develop an effective and reliable system for detecting AI-generated audio. As deepfake technology continues to evolve, the gap between synthetic and real speech narrows, making it crucial to design robust detection mechanisms. Conventional detection techniques, based on spectral analysis and hand-crafted feature engineering, have been found to be insufficient in detecting advanced deepfake audio. New developments in deep learning, especially Convolutional Neural Networks (CNNs), Dense Neural Networks (DNNs), and pre-trained models such as Wav2Vec, have been found to be effective in detecting minute anomalies in speech patterns. This research proposes to combine these cutting-edge methods into a single framework—TrueeTone: Audio Authenticity Detection—to improve the accuracy and efficiency of deepfake audio detection.

## 1.2 Problem Statement

The increasing sophistication of deepfake audio has made it difficult to verify the authenticity of speech in various applications. Current detection techniques often rely on handcrafted features or traditional machine learning models, which struggle with generalization across diverse datasets. Additionally, the growing accessibility of voice synthesis tools makes it easier for malicious actors to exploit deepfake technology for unethical purposes, such as impersonation scams, political propaganda, and misinformation campaigns. A major limitation of existing solutions is their dependency on either spectral analysis or manually extracted features, which do not always capture the deep patterns in AI-generated audio. As a result, there is an urgent need for a more efficient, scalable, and accurate detection system that can classify real and fake audio with high precision.

The TrueeTone project aims to address this problem by integrating deep learning models, including Dense Neural Networks (DNN), Convolutional Neural Networks (CNN), and the Wav2Vec model, to improve detection accuracy and robustness. The system seeks to provide an accessible and reliable tool for users to analyze audio authenticity effectively. Moreover, as deepfake technology evolves, attackers are continuously refining their methods to produce more realistic fake audio.

## 1.3 Objectives

The TrueeTone project is designed to develop a highly accurate and efficient audio authenticity detection system capable of distinguishing between AI-generated and real human speech. The key objectives of this research are as follows:

- To develop a deep learning-based framework that integrates multiple models for detecting AI-generated speech with high precision. This framework will leverage feature extraction techniques and multiple deep learning architectures to improve classification accuracy.
- To utilize MFCC and Wav2Vec feature extraction techniques for capturing both low-level and high-level speech representations, ensuring enhanced classification accuracy. These features will help models understand the intricate differences between real and AI-generated speech.
- To improve the generalization capabilities of audio detection models by training them on diverse datasets comprising both real and AI-generated speech. This ensures that the model can effectively detect deepfake audio in various environments and recording conditions.
- To implement a web-based user interface using Streamlit, allowing users to upload and analyze audio files easily. The interface will provide visualization tools like mel spectrograms to help users interpret the results.
- To evaluate model performance using metrics such as accuracy, precision, recall, and F1-score, ensuring the effectiveness of the proposed system. These performance measures will help identify the strengths and weaknesses of each model.
- To provide a real-time analysis mechanism for quick and accurate detection of deepfake audio in various practical applications. Real-time processing will enhance the usability of the system in security-sensitive environments.
- To explore hybrid model architectures that combine the strengths of different learning methods, improving robustness and adaptability. This approach will enhance the system’s ability to

handle emerging deepfake audio technologies.

- To enhance security and prevent misinformation by offering an effective solution for identifying manipulated audio content. This can help businesses, journalists, and organizations maintain trust in audio-based communications.

By achieving these objectives, this project aims to enhance security, trust, and reliability in digital audio communication, preventing misinformation and fraudulent activities. The development of a robust and efficient detection system will contribute significantly to the ongoing efforts in combating AI-generated voice manipulations.

## 1.4 Summary

The proliferation of deepfake technology has raised concerns about the authenticity of digital media, particularly in audio manipulation. As AI-generated voices become more convincing, there is an increasing need for detection systems that can differentiate between real and synthetic speech. The TrueeTone project is motivated by the growing risks associated with deepfake audio, such as misinformation and identity fraud, and seeks to develop a robust solution using deep learning. This chapter has outlined the motivation, problem statement, and objectives of the research. The motivation section highlights the ethical and security concerns posed by deepfake audio and the necessity for effective detection mechanisms. The problem statement identifies the current limitations of existing models, emphasizing the need for a scalable and accurate approach to detecting AI-generated speech. The objectives define the goals of the project, which include model integration, feature extraction, and real-time analysis. Through this research, TrueeTone aims to contribute to the field of audio authenticity detection by developing a multi-model system that leverages DNN, CNN, and Wav2Vec for improved performance. The project's unique approach integrates multiple deep learning techniques with advanced feature extraction, enhancing the ability to distinguish between human and AI-generated speech accurately. The need for robust deepfake detection solutions is evident, as AI-generated media continues to challenge the integrity of digital communication. By providing an automated, accurate, and accessible detection tool, this project aims to aid in the prevention of misinformation, identity fraud, and unethical media manipulation. The subsequent chapters will delve into the methodology, system design, implementation, and evaluation of the proposed approach, providing insights into the technical aspects and performance of the detection framework.

# CHAPTER 2
# LITERATURE SURVEY

# CHAPTER 2
# LITERATURE SURVEY

## 2.1 Introduction

The rise of deepfake technology has led to significant advancements in the field of synthetic media, allowing for the creation of highly realistic audio and video content. While this technology has many beneficial applications, such as in entertainment, gaming, and creative industries, its potential for misuse is alarming. One of the most dangerous forms of misuse is the creation of deepfake audio, which can convincingly mimic the voices of individuals, making it difficult to distinguish between real and synthetic content. This poses substantial risks in areas like identity theft, fraud, misinformation campaigns, and cybersecurity breaches. For instance, fake audio clips of executives' voices have been used to authorize fraudulent transactions, resulting in significant financial losses. Given the implications of deepfake audio on privacy, security, and trust, the need for effective detection methods has become critical. In response to these threats, numerous research studies have been published to develop detection systems capable of distinguishing between real and fake audio. These efforts focus on various methodologies, including traditional machine learning models, feature extraction techniques like MFCC (Mel Frequency Cepstral Coefficients), and advanced deep learning models, such as CNNs, self-supervised learning models, and multimodal detection systems that combine audio and video features. However, while many models show promise, they still face limitations in terms of accuracy, real-time application, and generalizability across diverse datasets. These challenges are compounded by the increasing sophistication of deepfake audio generation algorithms, which make subtle manipulations that are often difficult to detect using conventional methods. The papers reviewed in this chapter offer a comprehensive look at the current landscape of deepfake audio detection research. The methodologies they present provide useful insights but also reveal gaps that the "TrueeTone: Audio Authenticity Detection" project aims to address. This chapter will examine the various approaches proposed by different researchers, detailing the strengths and limitations of each method. It will also identify the key areas where these approaches fail to meet the requirements for real-world deepfake audio detection, particularly in terms of computational efficiency, generalization across datasets, and real-time performance. The review of existing literature will set the foundation for the development of the "TrueeTone" system, which aims to offer a more robust, scalable, and accurate solution for detecting deepfake audio.

## 2.2 Literature Survey

The current landscape of deepfake audio detection is shaped by a diverse range of research papers, each employing unique methodologies and innovative technologies to tackle the growing challenge of synthetic audio. These studies have collectively contributed to advancing detection techniques by exploring various strategies such as self-supervised learning, machine learning, and deep neural networks, as well as integrating both feature engineering and multimodal approaches. Each method offers distinct advantages, while also presenting specific limitations that highlight areas for further improvement. Several papers utilize self-supervised learning models like XLS-R and HuBERT to detect deepfake audio by leveraging pre-training on large datasets to extract robust features. These models demonstrate improved detection sensitivity and generalization across different types of synthetic speech. However, they often struggle with adaptability to new deepfake generation techniques, diverse languages, or varying accents, which restricts their real-world applicability. Other studies emphasize the use of machine learning models such as Support Vector Machines (SVM) combined with feature extraction techniques like Mel Frequency Cepstral Coefficients (MFCC). These methods excel in controlled environments but falter when faced with unstructured or noisy audio data, limiting their robustness in practical scenarios. Deep learning approaches that process raw audio data directly offer promising results by identifying subtle patterns within waveforms. While effective in detecting fake audio, these methods often demand significant computational resources, making them unsuitable for real-time deployment on devices with limited processing power. Hierarchical and explainable AI techniques provide interpretability and accuracy improvements but may lack scalability or sufficient detection precision for real-world use cases. Multimodal systems that combine audio and visual data further enhance detection capabilities but are less applicable to audio-only scenarios. These existing challenges underscore the need for a comprehensive solution like the "TrueeTone: Audio Authenticity Detection" project.

In [1], the study introduces a method for deepfake audio detection by integrating a self-supervised pre-trained model with a speaker-level classifier. The approach focuses on analyzing audio data to detect synthetic speech patterns at a speaker level. The dataset primarily consists of audio samples generated by various deepfake algorithms. The technique leverages the strengths of self-supervised learning, enhancing detection sensitivity. However, the method is highly sensitive to the type of deepfake generative algorithms used, reducing adaptability to new techniques. This limitation can be addressed by incorporating multiple detection models to improve accuracy and robustness.

In [2], a self-supervised model is utilized for deepfake audio detection, emphasizing the extraction of robust features from audio data. The dataset includes a variety of deepfake audio samples from different sources. The algorithm effectively generalizes across multiple deepfake types by relying on self-supervised pre-training. Despite this, the approach struggles when dealing with diverse languages and accents, reducing its universal applicability. Our project aims to overcome this by integrating multiple models that enhance performance across various languages and accents.

In [3], the research uses Mel Frequency Cepstral Coefficients (MFCC) for feature extraction combined with machine learning classifiers like Support Vector Machines (SVM) to detect deepfake audio. The dataset comprises structured audio samples, making it suitable for controlled environments. This method captures frequency characteristics effectively but faces challenges when handling unstructured or noisy audio data, limiting real-world application. By combining five models, our project aims to improve robustness and accuracy in noisy environments.

In [4], a deep learning approach processes raw audio waveforms directly for detecting deepfakes, eliminating the need for feature extraction. The dataset includes raw audio data from synthetic and real sources. This method shows promise by effectively capturing raw waveform patterns. However, it requires substantial computational resources, making real-time deployment difficult on devices with limited power. Our project addresses this drawback by optimizing detection models for efficiency and performance on resource-constrained devices.

In [5], the method employs a pre-trained self-supervised model to identify deepfake audio by learning audio representations without labeled data. The dataset covers various deepfake audio samples, and the approach improves detection accuracy. However, the model requires extensive fine- tuning on diverse datasets, limiting its real-world applicability when new data emerges. Our project mitigates this by integrating multiple models to adapt dynamically to new datasets, enhancing detection capabilities.

In [6], a multimodal system combines audio and visual data for detecting deepfakes involving both components. The dataset includes synchronized audio-visual samples. While effective for audio- visual deepfakes, this approach is less applicable to audio-only detection tasks. Our project focuses specifically on improving audio-only deepfake detection by leveraging multiple models to address various limitations in current techniques.

In [7], a survey reviews different methods for detecting deepfake audio, video, and multimodal content. The paper highlights strengths and weaknesses but lacks insights into real-time performance. The dataset diversity and detection speed are not thoroughly addressed. Our project improves upon this by building a system that ensures real-time detection performance with higher accuracy using a combination of five models.

In [8], explainable AI methods are used to detect deepfake audio by making detection processes interpretable. The dataset contains labeled deepfake audio samples analyzed by convolutional and recurrent neural networks. Although the approach is insightful, it achieves only 82.56% accuracy, which is insufficient for practical use. Our project enhances accuracy by combining multiple models to ensure higher detection reliability.

In [9], deep neural networks are applied to detect fake audio by analyzing amplitude patterns. The dataset consists of controlled audio samples, making the approach effective in laboratory conditions. However, the model struggles with real-world audio complexity, especially when background noise is present. Our project addresses this limitation by integrating diverse models to handle complex and noisy environments more effectively.

In [10], a fully automated end-to-end system detects deepfake audio using raw waveforms. The dataset includes various audio types, processed directly by a deep learning model. While effective, the method lacks flexibility in feature customization, making it difficult to adapt to new deepfake techniques. Our project overcomes this by incorporating multiple models to ensure adaptability and accuracy across evolving deepfake algorithms.

In [11], hierarchical representation learning and spectrogram features are used to detect fake audio. The dataset includes structured audio samples analyzed for temporal and spectral features. While improving accuracy, the approach struggles with generalization to new datasets. Our project addresses this by combining models that enhance generalization to diverse and unseen deepfake samples.

In [12], Mel-Frequency Cepstral Coefficients (MFCC) and Dynamic Time Warping (DTW) are applied for speaker recognition. These techniques extract spectral features from voice samples and align them temporally, making them suitable for clean and structured datasets. However, their performance significantly deteriorates in the presence of noise or distortion, as they are sensitive to environmental interference. This limits their applicability for deepfake detection in real-world settings, where audio is often noisy and unstructured. Our project addresses this by integrating noise-robust models that employ advanced Pre-processing and feature extraction techniques to maintain accuracy across varying conditions.

In [13], an autoregressive model generates raw audio using dilated convolutions, leveraging a dataset of high-quality speech samples. This method excels at producing realistic audio by modelling temporal dependencies in speech, but its computational intensity renders it unsuitable for real-time applications. Processing delays and high resource requirements restrict its deployment in practical scenarios where rapid detection is essential. Our project resolves this challenge by optimizing model architectures, incorporating lightweight convolutional layers and pruning techniques to reduce computational cost without sacrificing performance, enabling real-time detection.

In [14], feature engineering approaches combine MFCC and Principal Component Analysis (PCA) with machine learning classifiers like Support Vector Machines (SVM) and Random Forests to detect deepfake audio. While effective on structured datasets, this approach struggles with noisy, unstructured, or diverse audio inputs due to limitations in feature generalizability. Such constraints reduce its deployment potential in real-world conditions. Our project introduces a hybrid ensemble of five models, each specializing in different data characteristics. This ensemble approach ensures robust performance by leveraging complementary strengths, effectively handling noisy and heterogeneous datasets.

The implementation of TrueeTone: Audio Authenticity Detection is grounded in key findings from existing research on deepfake audio detection. Studies have demonstrated the effectiveness of feature extraction techniques like Mel Frequency Cepstral Coefficients (MFCC) in capturing relevant frequency-domain features of audio signals, making it a crucial component in our Dense Neural Network (DNN) and Convolutional Neural Network (CNN) models. While traditional machine learning-based classifiers, such as Support Vector Machines (SVM), have been used with MFCC, they struggle with real-world noisy data. Instead, deep learning-based approaches offer better adaptability and feature learning capabilities, motivating our use of CNNs and DNNs for classification tasks. Furthermore, literature highlights the growing prominence of self-supervised models like Wav2Vec, which learn robust audio representations without requiring labeled data. Recognizing its superior feature extraction capabilities for raw waveform analysis, TrueeTone incorporates the pre-trained Wav2Vec-960h-base model to enhance detection accuracy. Existing research also suggests that deepfake audio detection models often struggle with generalization, especially when tested on unseen data. To address this, TrueeTone integrates multiple models, ensuring robustness across different deepfake generation techniques. Another key consideration is real-time usability, as some deep learning models demand extensive computational resources, making deployment challenging. By optimizing model architectures and implementing a lightweight Streamlit-based front-end, TrueeTone provides an interactive, user-friendly interface that supports real-time audio uploads and analysis. The backend processing pipeline seamlessly integrates multiple models, ensuring comprehensive predictions while maintaining efficiency. By leveraging insights from prior research and integrating deep learning with advanced feature extraction techniques, TrueeTone aims to provide a reliable, scalable, and accessible solution for detecting deepfake audio, addressing challenges in media authenticity and misinformation prevention.

Table 2.1 Literature Survey comparison table

| **Sl. No.** | **Paper Title, Author, Publication details** | **Methodology Used** | **Drawbacks** |
|---|---|---|---|
| 1 | Audio Deepfake Detection with Self-Supervised XLS-R and SLS Classifier *Qishan Zhang, et al. ACM Multimedia, 2024* [1] | Integrates XLS-R with SLS module to enhance sensitivity for detecting AI-generated audio. | Sensitive to changes in generative algorithms. |
| 2 | Deepfake Audio Detection Using Self-Supervised Pre-training *Lanting Li et al. ICASSP, 2024* [2] | Self-supervised HuBERT model improves generalization for detecting fake audio. | Poor performance across diverse languages. |
| 3 | Deepfake Audio Detection via MFCC Features Using Machine Learning *Ameer Hamza, et al. IEEE, 2023* [3] | Uses MFCC feature extraction combined with machine learning models like SVM for deepfake detection. | Lower performance on unstructured data. |
| 4 | Deepfake Audio Detection Using Raw Audio and Deep Learning *Tushar Kapoor, et al. CONIT, 2023* [4] | Deep learning model processes raw audio data for fake audio detection. | Requires high computational resources. |
| 5 | Audio Deepfake Detection via Pre-trained HuBERT Model *Rui Liu, et al. ICASSP, 2023* [5] | Utilizes HuBERT self-supervised learning for improved fake speech detection. | Needs fine-tuning for diverse datasets |
| 6 | Audio-Visual Deepfake Detection System Using Multimodal Deep | Multimodal deep learning combining audio and video | Primarily focused on video |

| | | | |
|---|---|---|---|
| | Learning *Aman Parikh, et al. IEEE, 2023* [6] | features for detecting deepfakes. | deepfakes |
| 7 | Audio Deepfake Detection: A Survey *Zahra Khanjani, et al arXiv, 2023* [7] | Survey on deepfake detection methods, covering multiple deepfake domains. | Limited real-time detection insights. |
| 8 | Detecting Deepfake Voice Using Explainable Deep Learning Techniques *Suk-Young Lim, et al. MDPI, 2022* [8] | Utilizes explainable AI (XAI) with CNN and LSTM for deepfake voice detection, emphasizing interpretability. | Accuracy is only 82.56%, limiting suitability for real-world applications. |
| 9 | Deepfake Audio Detection Using Deep Neural Networks *Abu Qais et al. IEEE, 2022* [9] | Deep neural networks detect fake audio patterns, focusing on amplitude. | Accuracy drops with complex audio. |
| 10 | Fully Automated End-to-End Fake Audio Detection *Chenglong Wang, et al. arXiv 2021* [10] | Limited feature customizationn using wav2vec with optimized network structures. | Limited feature customization. |
| 11 | Fake Audio Detection Using Hierarchical Representations Learning and Spectrogram Features *Mohsan Ali, et al. ICRA, 2021* [11] | Limited dataset generalizationatures and LSTM for fake audio detection. | Limited dataset generalization. |
| 12 | Voice Recognition Using MFCC and DTW Techniques *Lindasalwa Muda, et al. ICASSP, 2021* [12] | MFCC for feature extraction and DTW for temporal sequence matching. | Performance degradation in noisy conditions. |

| | | | |
|---|---|---|---|
| 13 | WaveNet: A Generative Model for Raw Audio *Aäron van den Oord, et al. NeurIPS, 2020* [13] | Autoregressive model generates raw audio with dilated convolutions. | High computational cost. |
| 14 | Deepfake Audio Detection via Feature Engineering and Machine Learning *Farkhund Iqbal, et al. ICRA, 2021* [14] | Uses MFCC features and PCA for feature extraction with classifiers. | Limited on unstructured audio. |

## 2.3 Summary

The review of existing literature provides valuable insights into the various techniques and methodologies used for deepfake audio detection, but it also highlights several key challenges, While MFCC remains a widely used method, it struggles with unstructured and noisy data. The "TrueeTone" project will combine MFCC with more advanced techniques like CNN-based feature extraction and Wav2Vec to enhance accuracy and robustness. Many models, such as those by Qishan Zhang et al. (2024) and Rui Liu et al. (2023), struggle with generalizing across different datasets, particularly in diverse linguistic and environmental conditions. "TrueeTone" will focus on training models on diverse datasets to ensure better generalization. Several models, such as Tushar Kapoor et al. (2023), are computationally expensive, limiting their real-time applicability. The "TrueeTone" system will optimize models for efficiency and real-time performance, utilizing Gradio for a lightweight interface. Some models, like those by Abu Qais et al. (2022), show reduced accuracy in complex or noisy audio. By integrating multiple models, including Dense Neural Networks and pre-trained models like Wav2Vec, "TrueeTone" aims to improve accuracy even in challenging audio conditions.In conclusion, while existing research offers promising methods for deepfake audio detection, many of these systems are hindered by issues like poor generalization, high computational requirements, and reduced accuracy in complex or noisy scenarios. The "TrueeTone" project aims to address these gaps by developing a more robust, scalable, and efficient solution that can accurately detect deepfake audio in real-time applications.

# CHAPTER 3

# SYSTEM REQUIREMENT SPECIFICATIONS

# CHAPTER 3
# SYSTEM REQUIREMENT SPECIFICATIONS

## 3.1 Hardware Requirements

The hardware requirements for the TrueeTone: Audio Authenticity Detection system are crucial in ensuring efficient performance, seamless model execution, and real-time detection. The interface should display mel spectrograms and waveform plohigh-performance hardware to handle feature extraction, classification, and inference processes efficiently.

1. Processor: A multi-core processor with high computational power is essential. A minimum of an Intel Core i7 (10th Gen and above) or AMD Ryzen 7 or Apple M1 is recommended. For optimal performance, Intel Core i9 or Apple M1 is preferable to manage deep learning workloads effectively.
2. GPU: A powerful Graphics Processing Unit (GPU) is required to accelerate deep learning model training and inference. A minimum NVIDIA GTX 1660 Ti with 6GB VRAM is recommended. However, for better efficiency, an NVIDIA RTX 3080 or RTX 4090 with 12GB+ VRAM can significantly reduce training time and improve performance.
3. RAM: The deep learning models used in TrueeTone require substantial memory for processing large datasets. A minimum of 16GB RAM is needed, though 32GB or more is preferred for faster data handling and computation.
4. Storage: High-speed storage is necessary to store datasets, model checkpoints, and log files. A 512GB SSD is the minimum requirement, while 1TB SSD or NVMe SSD is recommended for faster read/write speeds and reduced data access latency.
5. Audio Interface: A high-quality sound card or audio interface is necessary for capturing real-time audio recordings accurately. A USB audio interface with 24-bit/96kHz support is ideal for ensuring high-quality audio input.
6. Internet Connection: A stable internet connection is required for downloading pre-trained models (such as Wav2Vec), accessing cloud-based APIs, and deploying the system on cloud-based infrastructures if needed.
7. Additional Peripherals: Basic peripherals, including a microphone (for real-time audio recording) and speakers/headphones for output verification, should be included to facilitate the testing and analysis of the system.

## 3.2 Software Requirements

The TrueeTone: Audio Authenticity Detection system depends on a robust set of software tools and frameworks to facilitate data pre-processing, deep learning model implementation, and user interface development. The software requirements are categorized into operating system, programming languages, libraries, frameworks, and deployment tools.

1. Operating System: The system is designed to run on Windows 10/11, Linux (Ubuntu 20.04+), and macOS. Linux is preferred for training deep learning models, as it provides better compatibility with NVIDIA CUDA for GPU acceleration. Windows and macOS are suitable for frontend development and deployment.
2. Programming Languages:

   Python 3.8+: Used for implementing machine learning and deep learning models, handling data pre-processing, and integrating APIs. JavaScript: If extending the project for a web-based frontend, JavaScript frameworks can be utilized.
3. Libraries and Frameworks:
   - TensorFlow/Keras & PyTorch: For deep learning model implementation.
   - Librosa & Torchaudio: For audio feature extraction.
   - Streamlit: To build the interactive user interface for real-time audio detection.
   - Scikit-learn & NumPy: For machine learning model training and data manipulation.
4. Development and Deployment Tools:
   - Google Colab & Jupyter Notebook: For model training and testing.
   - Ngrok: For deploying and exposing local servers online.
   - Streamlit Community Cloud: For deploying and exposing global servers online.

The choice of software tools ensures scalability, efficiency, and compatibility, making TrueeTone an adaptable and future-proof system.

## 3.3 Functional Requirements

Functional requirements define the core operations and system capabilities necessary for the successful execution of the TrueeTone: Audio Authenticity Detection system. These include:

1. Audio Input Handling:

   The system must accept audio files in common formats like .wav, .mp3, .flac for analysis. It must provide an option for live recording using a microphone.

2. Feature Extraction and Pre-processing:

   The system should extract Mel-Frequency Cepstral Coefficients (MFCC), spectrograms, and waveform features. Wav2Vec must process raw audio for robust feature representation.

3. Deep Learning-Based Analysis:

   The system must process the extracted features through multiple deep learning models, including DNN, CNN, and Wav2Vec. The models should classify whether the input audio is genuine or deepfake.

4. Model Comparison and Performance Metrics:

   The results from multiple models should be compared and displayed. Performance metrics such as accuracy, precision, recall, and F1-score should be provided.

5. User Interface and Interaction:

   A user-friendly web application should allow users to upload or record audio and receive results.

   The interface should visually display Mel spectrograms and detection confidence scores.

6. Real-Time Detection and Response:

   The system should ensure fast inference times to support real-time detection. Results must be interpretable and displayed with explanations.

7. Secure and Scalable Model Deployment:

   The system should be deployable using cloud or local hosting. The architecture must be scalable to accommodate future enhancements.

By implementing these functional requirements, TrueeTone ensures accuracy, efficiency, and accessibility for detecting deepfake audio.

## 3.4 Non-Functional Requirements

Non-functional requirements define the quality attributes that the TrueeTone system must meet to ensure optimal performance, usability, and scalability.

1. Performance and Scalability:

The system should process audio files and return results within a few seconds to support real-time usability. The architecture must support scalability, allowing future integrations with new AI models or additional functionalities.

2. Reliability and Accuracy:

   The detection models must achieve high accuracy rates (above 90%) for deepfake identification. The system must provide consistent performance across different types of audio data.

3. Security and Data Privacy:

   User-uploaded audio files must be securely processed without unauthorized access. The system must comply with data protection policies to prevent misuse of sensitive voice data.

4. Usability and User Experience:

   The UI must be intuitive and responsive, making it accessible for technical and non-technical users. A clear and informative display of detection results, including spectrogram visualizations, should be included

5. Compatibility and Portability:

   The system should be deployable on Windows, Linux, and cloud platforms. It must be lightweight enough to run on personal computers with standard specifications.

6. Maintainability and Extensibility:

   The project codebase should be modular and well-documented, allowing future updates. The system must allow integration with future AI models for improved detection capabilities.

By adhering to these non-functional requirements, TrueeTone ensures optimal performance, high usability, and security, making it a robust and scalable deepfake audio detection system.

## 3.5 Summary

System Requirement Specifications for TrueeTone: Audio Authenticity Detection, ensuring that the project meets both functional and performance expectations. This chapter defines the hardware, software, functional, and non-functional requirements necessary for the successful implementation and deployment of the system. The hardware requirements emphasize the need for a high-performance CPU and GPU to support deep learning models like Wav2Vec, CNN, and DNN. During development, a system with at least 32GB RAM, an NVIDIA RTX GPU, and 1TB SSD storage is recommended for efficient model training. For deployment, a lighter system with 16GB RAM, SSD storage, and cloud integration options ensures smooth inference and real-time processing. The software requirements cover the necessary tools and frameworks, including Python, TensorFlow, PyTorch, Streamlit, Librosa, and Scikit-learn. The system runs on Windows, Linux, and macOS and can be deployed using Google Colab, AWS, or local servers. Additionally, Ngrok is used for exposing the local Streamlit-based frontend for user interaction. The functional requirements define the key operations of the system, such as audio file input, feature extraction using MFCC and spectrograms, deep learning-based classification, and real-time detection. The user interface provides an interactive experience, allowing users to upload or record audio, visualize spectrograms, and receive classification results.The non-functional requirements focus on performance, accuracy, scalability, security, and user experience. The system ensures fast inference times, data privacy, high accuracy rates (above 90%), and seamless compatibility across devices.

# CHAPTER 4
# SYSTEM DESIGN

# CHAPTER 4

# SYSTEM DESIGN

## 4.1 High-Level Design

The high-level architecture of the TrueeTone: Audio Authenticity Detection system is modular, involving a sequence of processes that convert raw audio input into a classification output—either AI-generated (Fake) or Human-generated (Real). The system comprises three core stages: Input & Preprocessing, Model Inference, and Result Display, all orchestrated through a Streamlit-based web interface that serves as the frontend for user interaction. At the input level, users can either upload audio files (.wav) or record live audio through the interface. Once an audio input is received, the system determines the appropriate preprocessing path based on the model being used. For the DNN and CNN models, MFCC (Mel Frequency Cepstral Coefficients) are extracted from the audio file, transforming it into a 2D numerical representation that captures the essential spectral features of speech. For the Wav2Vec model, the raw waveform is passed directly into the pretrained Wav2Vec-960h-base model without MFCC transformation, leveraging its self-supervised learning capabilities.

In the Model Inference stage, the audio input is passed through all three models in parallel. Each model generates a prediction independently. The DNN and CNN models, developed from scratch, are trained using supervised learning on a labeled dataset of AI and human audio samples. These models are compiled with the Adam optimizer, categorical cross-entropy loss, and include several hidden layers with ReLU activation, dropout for regularization, and batch normalization (in CNN). On the other hand, the Wav2Vec model is fine-tuned from a Hugging Face transformer pipeline and gives the most accurate prediction due to its advanced contextual understanding. The Result Display stage is where the output of each model is shown.

## 4.2 Design Considerations

The design of TrueeTone involved several key decisions to ensure the system was accurate, efficient, and user-friendly. The choice to use multiple models—DNN, CNN, and Wav2Vec—was intentional, providing an ensemble-like framework to validate predictions across architectures. Each model was carefully built and trained based on the nature of audio data and how different network types perform on frequency-based features. For example, the CNN's ability to detect spatial patterns was particularly effective in extracting locality-based information from MFCCs, while the DNN was implemented with deeper layers and dropout regularization to generalize well on moderately sized datasets. An important design decision was to maintain modularity. Each model is encapsulated in its own script and training pipeline (dense_training.py, cnn_training.py, and feature_extraction_wav2vec.py). This allows for future updates or replacement of models without affecting the rest of the system. Additionally, the entire interface and model execution is centralized within the trail7.py script, allowing seamless execution from data input to final output within a single app instance. The interface was designed with simplicity and accessibility in mind. Streamlit enables a quick and dynamic UI setup where users can interact with the system without needing any coding knowledge. The ability to record audio or upload existing files caters to both real-time and offline scenarios. Visual feedback through mel spectrograms aids in transparency, allowing users to visually inspect the frequency content of their audio.

### 4.2.1 Assumptions and Dependencies

Several assumptions underpin the system's design. First, it is assumed that all input audio will be in a processable format (ideally .wav) and of sufficient quality (i.e., minimal background noise, clear speech). Although there is basic audio format validation in place, no deep noise reduction or voice enhancement techniques are implemented. It is also assumed that the models have been sufficiently trained and are not prone to overfitting; therefore, model selection and validation during training are critical. The system depends on several Python libraries and frameworks for both machine learning and interface development. These include:

- Librosa and NumPy for feature extraction,
- TensorFlow and Keras for building and training DNN and CNN models,
- Transformers and Torch for loading the pretrained Wav2Vec model,
- Matplotlib and Streamlit for visualization and UI, and
- Ngrok for tunneling and public web access.

In addition, the Wav2Vec model relies on the availability of pretrained weights from Hugging Face, and the internet connection must be active during initialization if models are not locally cached.

### 4.2.2 Goals and Constraints

The main goal of TrueeTone is to provide a reliable, lightweight, and accessible tool for detecting whether a piece of audio is AI-generated or human-generated. With the growing accessibility of deepfake voice technologies, the system is designed to address a crucial need in digital media authenticity and verification. Another goal was to build a solution that can run in constrained environments—hence the decision to host it via Google Colab and deploy through Ngrok rather than a dedicated server. The system faced a few constraints during development:

- Computational limitations: Training CNN and DNN models was done in Colab, and model complexity had to be balanced to fit GPU memory constraints.
- Data diversity: Audio datasets needed to include a balanced mix of AI-generated and real speech to avoid bias. Any skew could reduce real-world accuracy.
- Latency: Due to model size and inference time (especially Wav2Vec), optimizing runtime without sacrificing accuracy was a constraint during model and UI integration.
- Scalability: Since the app is hosted on a temporary tunnel, it's not intended for high-traffic or concurrent multi-user environments.

Despite these limitations, the system meets its primary objectives of detecting fake audio with high accuracy and offering a user-friendly platform for interaction. Its modular design, ease of deployment, and solid model performance make it a strong foundation for future extensions, such as speaker verification or emotional tone analysis.

## 4.3 System Architecture

The system architecture of TrueeTone: Audio Authenticity Detection is visually represented in Figure 4.1 System Architecture, which outlines the structured, multi-layered approach adopted in the project for end-to-end audio authenticity verification. The architecture is divided into key functional blocks—Input Layer, Data Preprocessing, Feature Processing, Model Processing, Integration Layer, and Output Layer—each responsible for a distinct part of the pipeline, enabling modularity, flexibility, and accuracy. As shown in the figure, the process begins at the Input Layer, where users can either upload .wav files or record live audio. This audio data then flows into the Data Preprocessing stage, where it undergoes standardization through audio loading and resampling, ensuring uniform format and sample rate for all inputs.

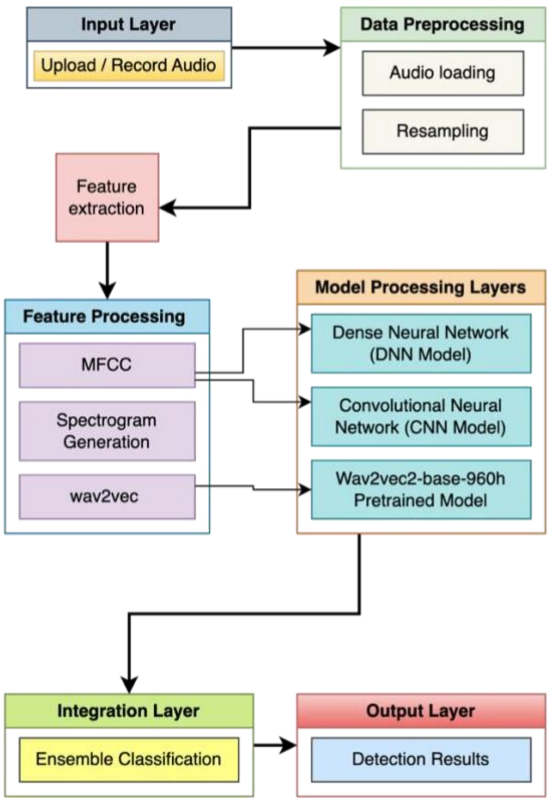


Figure 4.1 System Architecture Diagram

Within the Feature Processing block, three types of features are extracted based on the model requirements. MFCCs are computed for the DNN and CNN models, spectrograms are generated for visualization, and raw waveform features are prepared for the Wav2Vec2 model. This enables the system to exploit both handcrafted and deep-learned representations of audio data. These processed features are then routed to the Model Processing Layers, which include three models: the custom-built Dense Neural Network (DNN), a Convolutional Neural Network (CNN) trained on MFCCs, and a Wav2Vec2-base-960h pretrained model from Hugging Face that directly handles raw waveform inputs. Post individual model inference, the results are unified through the Integration Layer, where an ensemble-like logic selects or highlights the strongest prediction—typically from Wav2Vec due to its robustness. Finally, the Output Layer displays the classification result (AI or Human) on the Streamlit interface, along with visual aids like the mel spectrogram, giving users both decision and insight. This architecture ensures an efficient, transparent, and accurate detection pipeline for modern deepfake audio verification.

## 4.4 Data Flow Diagram

### 4.4.1 Data Flow Diagram (Level 0)

The Level 0 Data Flow Diagram (DFD) of the TrueeTone system provides a high-level overview of the entire architecture shown in Figure 4.2, where the system is represented as a single process block. This diagram focuses on how the system interacts with external entities, mainly the user, without exposing any of the internal mechanics. In this context, the user provides input to the system in the form of an audio file—either recorded live or uploaded in .wav format. This input is then processed within the central Audio Analysis System, which handles feature extraction, analysis, and classification. Once the processing is completed, the system generates an output prediction indicating whether the audio is AI-generated or human-recorded. The diagram also shows a supporting module that aids in extracting audio features such as MFCCs or raw waveform analysis. The purpose of this Level 0 DFD is to define the boundary between the system and external actors, and to describe the fundamental data interactions: audio input from the user, internal processing, and the final prediction output delivered back to the user in a clear and concise format.

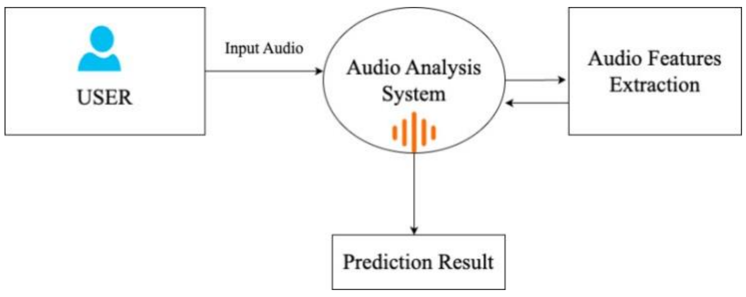


Figure 4.2 Data Flow Diagram (Level 0)

### 4.4.2 Data Flow Diagram (Level 1)

The Level 1 Data Flow Diagram visually represented in Figure 4.3, dives deeper into the internal structure of the TrueeTone system, decomposing the central system block from Level 0 into more granular subcomponents. This diagram illustrates the specific modules responsible for handling each stage of the audio analysis pipeline. It begins with the Pre-processing Module, which cleans the audio signal by resampling, normalizing, and optionally removing silence. Next, the data moves to the Feature Extraction Module, where MFCCs and mel spectrograms are generated for traditional models, while raw waveform data is passed to the Wav2Vec model. These extracted features are then fed into multiple trained models—namely, a Dense Neural Network (DNN), a Convolutional Neural Network (CNN), and a transformer-based Wav2Vec model. Each model independently performs classification and generates a prediction. These predictions are then passed to the Model Integration (Ensemble Classification) module, which combines the outputs—either through voting, confidence ranking, or prioritizing Wav2Vec's superior accuracy—to generate the final decision. The Results Output module then displays this final result along with visual aids like the mel spectrogram. Additionally, during model development, training data is pulled from a labeled dataset and goes through steps like splitting and resampling before feeding into the training phase. This detailed breakdown illustrates how audio input is transformed into an authenticity label through a sequence of structured and collaborative processes.

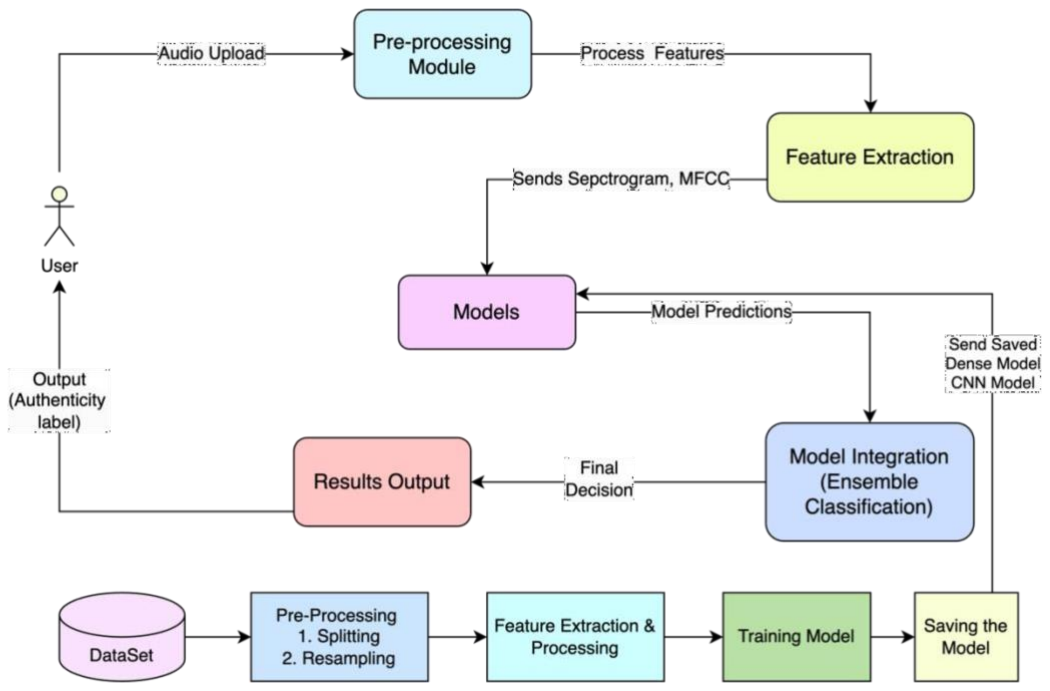


Figure 4.3 Data Flow Diagram (Level 1)

## 4.5 Activity Diagram

The activity diagram for the TrueeTone project illustrates the sequential flow of actions involved in analyzing audio authenticity. It begins when the user uploads an audio file, which initiates the preprocessing phase. During this step, the system performs necessary operations like data resampling, normalization, and trimming to ensure the audio input is in a suitable format for analysis. Following preprocessing, the next phase is feature extraction, where the system derives critical audio characteristics such as Mel Frequency Cepstral Coefficients (MFCCs) and Mel Spectrograms. These features are essential for identifying patterns and structures within the audio signal that may indicate whether it is human-generated or AI-generated. Once features are extracted, they are fed into three separate machine learning models: a Convolutional Neural Network (CNN) that interprets spectrograms as images, a Dense Neural Network (DNN) that processes MFCC feature vectors, and a transformer-based Wav2Vec model that operates directly on raw waveform data. Each model independently evaluates the audio and generates a prediction. These predictions are then passed to the ensemble decision module, where a majority voting or confidence-based integration strategy is used to finalize the authenticity classification. The final output—labeling the audio as either "Real" or "AI-generated"—is displayed to the user through the front-end interface, completing the activity flow. This diagram as shown in Figure 4.4, emphasizes the logical and modular nature of the system, ensuring that each component functions in a streamlined and cohesive manner.

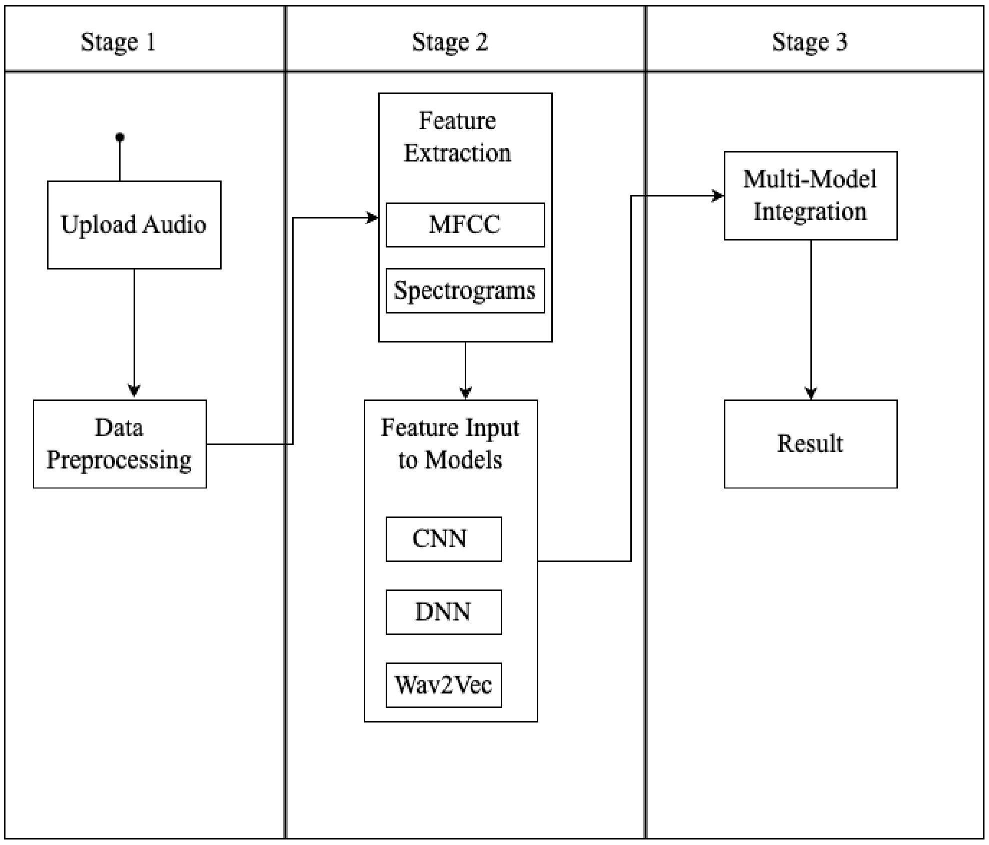


Figure 4.4 Activity Diagram

## 4.6 Use-Case Diagram

The use-case diagram of TrueeTone defines the interaction between the system and two key actors: the User and the Admin. The use-case diagram effectively outlines these interactions and presents a high-level overview of the functionalities provided to each actor, reflecting the complete scope of operational and maintenance interactions within the TrueeTone and feature extraction, where MFCC and spectrogram features are automatically extracted without requiring user input. The system then classifies the audio using the integrated models (CNN, DNN, and Wav2Vec), followed by the generation and display of the authenticity result. On the other hand, the Admin is a secondary actor responsible for system maintenance and improvements. The Admin is associated with functions such as retraining the models with updated or extended datasets and managing the data pipeline by updating the dataset as needed. This ensures that the system remains adaptive and up-to-date with new types of AI-generated audio. The use-case diagram visually presented in Figure 4.5, effectively outlines these interactions and presents a high-level overview of the functionalities provided to each actor, reflecting the complete scope of operational and maintenance interactions within the TrueeTone system.

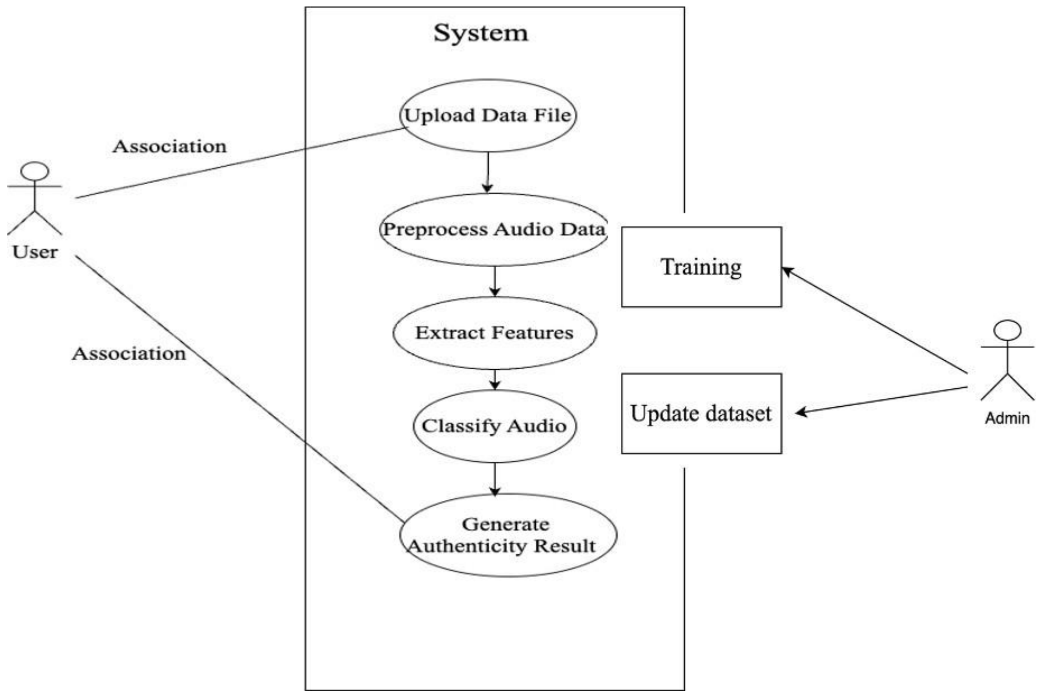


Figure 4.5 Use-Case Diagram

## 4.7 Sequence Diagram

The sequence diagram, refer to Figure 4.6, Provides a dynamic view of the interaction among various system components during the execution of the audio authenticity detection process. It captures the real-time flow of messages between the user, the interface, and the backend modules to achieve the final result. The sequence initiates when the User interacts with the Streamlit Front-End Interface by uploading an audio file or recording one in real time. This request is captured and passed to the Pre-processing Module, where audio is resampled, normalized, and trimmed to ensure consistency across the dataset. After preprocessing, the audio is forwarded to the Feature Extraction Module, which computes both Mel Frequency Cepstral Coefficients (MFCCs) and Mel Spectrograms. These extracted features are then sent to three separate classification models: the Dense Neural Network (DNN), which takes MFCC vectors; the Convolutional Neural Network (CNN), which interprets spectrogram images; and the Wav2Vec model, which processes the raw waveform directly using a pre-trained transformer architecture. Each model independently processes the input and returns a prediction score indicating whether the input is AI-generated or authentic. These prediction results are then passed to the Model Integration Module, where an ensemble decision-making approach—such as majority voting or confidence-based weighting—is applied to consolidate the outputs of the individual models. The final decision is then forwarded to the Result Output Module, which formats and displays the result on the Streamlit front end, indicating the authenticity status of the uploaded audio. This diagram not only reflects the sequential interaction between all internal components but also ensures clarity in how data flows and decisions are made step-by-step. It reinforces the modular yet integrated architecture of TrueeTone, where preprocessing, analysis, and prediction operate in harmony to deliver robust audio authenticity detection.

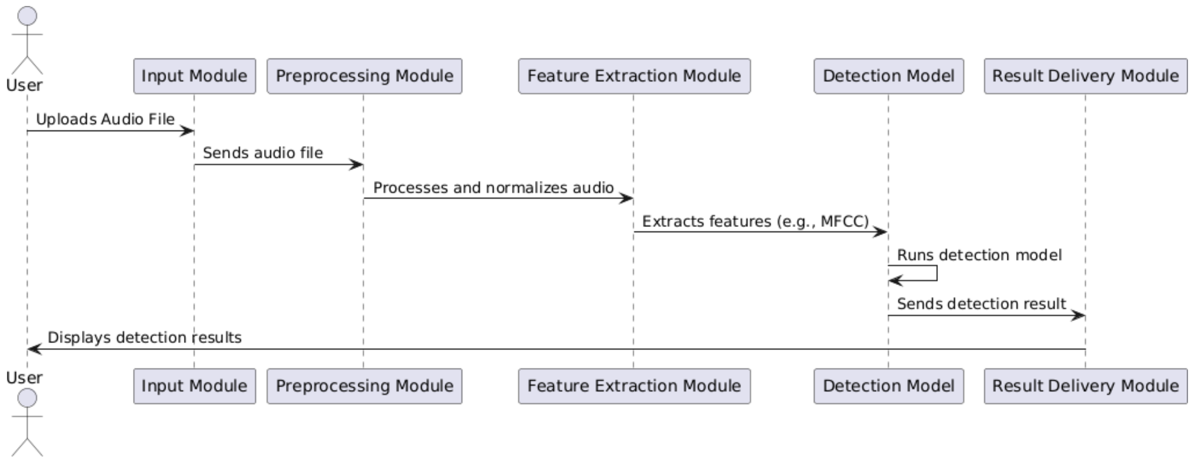


Figure 4.6 Sequence Diagram

## 4.8 List of Modules

The proposed system for audio authenticity detection consists of multiple integrated modules that work in a pipeline to process input audio, extract meaningful features, apply machine learning models, and deliver results. Below is the list of core modules and their respective descriptions:

1. Input Module:

   This module provides the interface for users to upload their audio files into the system. It supports audio formats such as .wav and handles file validation. This is the first point of interaction between the user and the system, and it ensures the audio file is appropriately formatted for further processing.

2. Preprocessing Module:

   The preprocessing module is responsible for preparing the audio data for analysis. It normalizes the audio signal, removes noise, and performs operations like resampling or trimming to ensure consistency. This step is critical to improve model performance and to reduce the effect of any audio artifacts or quality issues.

3. Feature Extraction Module:

   Once the audio is preprocessed, the Feature Extraction Module converts it into representations that machine learning models can interpret. It extracts Mel-Frequency Cepstral Coefficients (MFCC), spectrograms, or other time-frequency domain features, which serve as the input to various models like CNN, DNN, or Wav2Vec.

4. Detection Model Module:

   This is the core analysis module where machine learning models are applied to classify whether the given audio is authentic or manipulated. It includes:

5. Convolutional Neural Network (CNN):

   Focuses on spatial patterns in spectrograms.

6. Deep Neural Network (DNN):

   Uses extracted features for learning complex patterns.

7. Wav2Vec Model:

   A pre-trained model that processes raw audio directly for enhanced detection.

8. Result Delivery Module:

   The final module consolidates the predictions from all models using a Multi-Model Integration strategy. It compares outputs, highlights the best-performing model (often Wav2Vec), and displays the final authenticity result to the user in an intuitive format. It may also generate a downloadable report for record-keeping.

## 4.9 Dataset Description

The dataset used in the TrueeTone: Audio Authenticity Detection project is a carefully curated collection of audio samples divided into two distinct classes: real (human-spoken) audio and fake (AI-generated) audio. The primary goal of this dataset is to enable effective binary classification by training machine learning and deep learning models to differentiate between authentic and synthetic speech patterns. The real audio samples are sourced from publicly available speech corpora such as LibriSpeech, which contains clean and high-quality voice recordings of human speakers reading passages from books. These recordings vary in speaker accents, intonations, and pace, thus offering a wide spectrum of natural human speech characteristics. The inclusion of such diversity ensures that models trained on this dataset can generalize well to different speaking styles and avoid overfitting to a narrow voice pattern.

On the other hand, the fake audio samples are generated using advanced text-to-speech (TTS) synthesis engines, such as Google TTS, Microsoft Azure TTS, and Amazon Polly, as well as AI voice cloning models available on platforms like Descript's Overdub or Murf.ai. These AI-generated clips attempt to mimic natural human voices, but often exhibit anomalies in waveform patterns, unnatural rhythm, and mechanical tone modulation, which the system learns to detect. The dataset is split into training, validation, and test sets, maintaining a balanced distribution of both real and fake audio across each partition. Preprocessing involves converting all audio clips to a uniform sampling rate (typically 16 kHz), trimming silence, normalizing amplitudes, and converting to mono-channel format to reduce model complexity. This ensures compatibility across all the feature extraction pipelines, such as MFCC generation, Mel Spectrogram computation, and raw waveform processing for Wav2Vec. To improve robustness and reduce bias, data augmentation techniques such as noise addition, pitch shifting, and speed variation are selectively applied to the training set. These techniques help simulate real-world audio conditions and make the model resilient to background interference and speaker variability. Overall, this dataset serves as a foundational element for training, testing, and validating the ensemble of models used in TrueeTone. Its richness in variation and controlled class distribution ensures high performance, generalizability, and reliability of the system in identifying AI-generated speech.

## 4.10 Summary

Chapter 4 outlines the structural and functional design of the TrueeTone system. It begins with a high-level architecture that integrates three core models—DNN, CNN, and Wav2Vec—for audio authenticity detection. The design considerations highlight key goals such as accuracy, scalability, and real-time performance, along with assumptions like dataset availability and processing capacity. Detailed Data Flow Diagrams (Level 0 and 1) map the system's internal processes from audio input to final prediction. The Activity and Use-Case Diagrams explain both operational flow and user- system interactions. Additionally, the Sequence Diagram details communication between modules, while the list of model modules shows how each classifier contributes uniquely. Finally, the dataset description emphasizes its balanced nature and rich diversity, ensuring the system is capable of distinguishing between real and AI-generated speech effectively.

# CHAPTER 5
# IMPLEMENTATION

# CHAPTER 5

# IMPLEMENTATION

## 5.1 Introduction

The implementation phase of TrueeTone: Audio Authenticity Detection involved developing a robust system for classifying audio as either AI-generated (fake) or human-recorded (real). This implementation integrates multiple deep learning models alongside a user-friendly web interface that enables real-time testing, analysis, and visualization of audio authenticity. The system leverages deep learning techniques, signal processing, and web-based deployment to ensure accessibility and reliability. The implementation process was divided into two key components, Model Training & Development that Involves Pre-processing audio data, extracting relevant features, training deep learning models, and fine-tuning them for optimal performance. Secondly Front-End Integration – Focuses on developing a user-friendly web interface using Streamlit, enabling users to upload, record, analyze, and visualize audio classification results. The backend consists of three core deep learning models, Dense Neural Network (DNN) with MFCC-based feature extraction. Convolutional Neural Network (CNN) leveraging convolutional layers for improved feature detection.Pretrained Wav2Vec Model, which extracts deep speech features directly from raw audio. Each of these models has been trained on AI-generated and human-recorded speech datasets. The models process the input using different architectures and feature extraction techniques, ensuring a comprehensive analysis of the authenticity of the given audio sample. In the DNN model, MFCC (Mel-Frequency Cepstral Coefficients) are extracted from the raw audio and passed through multiple fully connected layers with ReLU activations. The training strategy involves balancing the dataset using RandomOverSampler, optimizing the model using the Adam optimizer and using binary cross-entropy loss to improve classification accuracy. The model is trained for 75 epochs with a batch size of 2, ensuring that small variations in data are effectively captured. The CNN model builds upon MFCC features as well but processes them through convolutional layers with 1×1 kernel filters to identify intricate frequency patterns. It employs two convolutional layers with 32 and 64 filters, followed by fully connected layers with a dropout mechanism to prevent overfitting. The CNN is trained for 50 epochs with an early stopping mechanism, ensuring optimal convergence. The Wav2Vec model, on the other hand, does not require handcrafted feature extraction like MFCCs. It directly processes the raw waveform, extracting deep hierarchical speech representations. The classification is performed based on entropy-based analysis, where a higher entropy value indicates human speech, while a lower entropy suggests AI-generated audio. This model provides the most accurate classification and is prioritized in decision-making. The front-end interface is developed using Streamlit, allowing users to upload or record audio, view Mel Spectrogram visualizations, and receive predictions from all three models.

## 5.2 Programming Language Selection

The selection of a programming language is a crucial aspect of any machine learning and web-based application. For the implementation of TrueeTone: Audio Authenticity Detection, Python was chosen as the primary programming language. The decision to use Python is based on its extensive library support, ease of use, and strong community backing, making it ideal for machine learning, deep learning, and web application development.

### 5.2.1 Why Python?

Python is the most popular language in AI, machine learning, and deep learning applications due to the following reasons:

- Rich ecosystem of libraries: Python provides powerful libraries such as TensorFlow, PyTorch, Librosa, Torchaudio, and Scikit-learn, which are essential for training deep learning models and handling audio data.
- Ease of integration: Python integrates well with Google Colab, making it easier to train models on GPU-supported cloud environments.
- Extensive community support: The vast Python community ensures continuous improvements, bug fixes, and support for complex implementations.
- Simplified syntax and readability: Python’s easy-to-read syntax enables efficient collaboration and debugging.
- Efficient deployment: Python-based applications can be easily deployed using frameworks like Streamlit and Flask, simplifying the transition from research to real-world application.

### 5.2.2 Programming Language Justification for Each Component

Python was selected for different components of the TrueeTone project due to its efficiency in handling data preprocessing, model training, and web deployment.

1. **Model Training and Development**

   - Dense Neural Network (DNN) Implementation: The DNN model was built using TensorFlow and Keras, which provide high-level APIs for developing neural networks. The feature extraction process was handled using Librosa, a specialized Python library for audio analysis. The dataset balancing was achieved using imbalanced-learn's RandomOverSampler, ensuring fair training. Training was optimized using the Adam optimizer, a built-in function in TensorFlow.
   - Convolutional Neural Network (CNN) Implementation: CNN was implemented using TensorFlow/Keras, taking advantage of its built-in support for convolutional layers. Audio preprocessing was performed using Librosa, where MFCC features were extracted. Model evaluation was carried out using Matplotlib and Scikit-learn's classification metrics, allowing for visualization of loss, accuracy, and confusion matrices.
   - Wav2Vec Pretrained Model: The Wav2Vec model was implemented using Facebook's Hugging Face library, which provides a seamless interface for pretrained speech models. The model leverages Torchaudio, a PyTorch-based library that enables efficient feature extraction from raw audio.

2. **Front-End Development**

   The front-end of TrueeTone was developed using Streamlit, a Python-based web framework known for its simplicity and rapid deployment capabilities. Streamlit was chosen due to:

   - Minimal coding requirements: Unlike Flask or Django, Streamlit requires fewer lines of code to develop interactive UIs.
   - Built-in support for visualization: TrueeTone uses Streamlit's charting functions to display Mel Spectrograms and MFCC features.
   - Seamless model integration: Python-based models can be directly loaded and executed in Streamlit without the need for an external API.
   - Lightweight deployment: The final application is deployed via Ngrok in Google Colab, ensuring a smooth experience for end-users.

3. **Audio Processing and Feature Extraction**

   Python's Librosa and Torch audio libraries were used for processing audio files. The decision to use these libraries is based on their efficiency in:

- Extracting MFCC features.
- Handling multiple audio formats (WAV, MP3, OGG, etc.).
- Generating Mel Spectrograms for visualization.
- Pre-processing raw waveform data for deep learning models.

4. **Report Generation and Deployment**

   To enhance usability, TrueeTone generates downloadable reports using FPDF, a Python-based PDF generation library. This allows users to receive detailed classification results, model predictions, and visualizations in a structured document format. For deployment, Pyngrok was used to create a publicly accessible link via Google Colab, enabling real-time testing of the application.

### 5.2.3 Conclusion

The decision to use Python for implementing TrueeTone was driven by its powerful machine learning ecosystem, efficient audio processing capabilities, and seamless integration with web-based frameworks. By leveraging Python’s rich library support and deep learning frameworks, TrueeTone successfully achieves high accuracy in detecting AI-generated speech while ensuring a smooth user experience through an interactive Streamlit interface.

## 5.3 Platform/Framework

The TrueeTone: Audio Authenticity Detection system is designed using a combination of Python-based deep learning frameworks and web application development tools. The system consists of a backend for model training and inference and a front-end for user interaction and real-time audio analysis. The implementation leverages TensorFlow, PyTorch, Librosa, Streamlit, and Ngrok, ensuring a robust and efficient pipeline for detecting AI-generated and human-recorded audio.

### 5.3.1 Backend Frameworks and Libraries

The backend is responsible for feature extraction, model training, and inference. Three models are implemented for classification. Each model uses different deep learning frameworks and signal processing tools, which are described below:

1. **TensorFlow and Keras for DNN and CNN**

   Both the Dense Neural Network (DNN) and CNN models are implemented using TensorFlow and Keras, providing a high-level API for defining and training deep learning architectures.

   - DNN Model (dense_training.py) uses MFCC (Mel Frequency Cepstral Coefficients) features extracted with the Librosa library. TensorFlow's Sequential API is used to stack multiple dense layers with ReLU activation functions and dropout regularization to prevent overfitting. The Adam optimizer is utilized to fine-tune the learning process.
   - CNN Model (cnn_training.py) follows a similar approach but introduces convolutional layers that capture spatial patterns from MFCC features, making the model more effective in audio classification tasks.

2. **PyTorch for Wav2Vec Model**

   The feature_extraction_wav2vec.py script leverages Facebook's Wav2Vec 2.0 pretrained model using the Transformers library from Hugging Face. PyTorch is used as the deep learning framework for extracting embeddings from raw audio waveforms. Unlike the MFCC-based models, Wav2Vec works directly on waveforms, eliminating the need for handcrafted feature extraction.

   - The extracted embeddings are passed to a classification head, implemented using PyTorch's Linear layers.
   - Fine-tuning is conducted using AdamW optimizer and CrossEntropy loss, enabling the model to classify between AI-generated and human-recorded audio effectively.

3. **Librosa for Audio Preprocessing**

   The Librosa library is a critical component for preprocessing audio data. It extracts MFCC features for both the DNN and CNN models. Additionally, it handles sampling rate conversion, waveform normalization, and audio augmentation techniques like time-stretching and pitch shifting to enhance model generalization.

4. **Numpy and Pandas for Data Handling**

   Throughout the pipeline, NumPy is used for array operations, ensuring efficient feature storage and manipulation. Pandas is utilized for dataset management, particularly when handling metadata such as labels and file paths.

### 5.3.2 Front-End Development and Integration

The TrueeTone project employs Streamlit to create an interactive web-based front-end, designed for intuitive user interaction with audio analysis. This interface enables users to upload or record audio, which is then visually represented as a Mel spectrogram, facilitating a clear understanding of the audio's frequency components. The core functionality involves the integration of pre-trained machine learning models – Wav2Vec, CNN, and DNN – to provide real-time predictions. Streamlit's selection is driven by its ease of use, rapid deployment capabilities, and native support for multimedia content, making it ideal for this application. The system highlights the model with the highest confidence score, ensuring users receive the most reliable prediction.

For model integration, TensorFlow's Keras API is utilized for the DNN and CNN models, while PyTorch's Transformer pipeline handles the Wav2Vec model. The audio input is processed using Librosa, converting it into suitable formats. MFCC features are extracted for the DNN and CNN models, and raw waveforms are used for Wav2Vec. The models then generate classification probabilities, which are presented to the user in a clear, accessible format, along with corresponding confidence scores. To enable external access and facilitate testing across diverse devices, Ngrok is implemented to create a secure public URL. This eliminates the need for complex server configurations, streamlining the deployment process and allowing for seamless user interaction regardless of location.

## 5.4 Tools

The development and implementation of TrueeTone: Audio Authenticity Detection required a combination of various tools, frameworks, and libraries to ensure efficiency, accuracy, and seamless functionality. The system integrates machine learning models with a web-based interface to detect whether an uploaded or recorded audio file is human-generated or AI-generated. The choice of tools was driven by the need for deep learning capabilities, signal processing, and a user-friendly front-end for real-time inference. For building and training the models, Python was the primary programming language due to its extensive support for machine learning and signal processing libraries. The training of the Dense Neural Network (DNN) and Convolutional Neural Network (CNN) relied on TensorFlow and Keras, which provided high-level APIs for defining, training, and optimizing deep learning architectures. The DNN model was trained using MFCC (Mel-Frequency Cepstral Coefficients) feature extraction, which was implemented using the Librosa library. Librosa is widely used for audio analysis and feature extraction, making it a crucial tool for processing audio data before feeding it into the neural network.

The Wav2Vec 960h base model, a pre-trained transformer-based model from Facebook AI, was integrated using the transformers library provided by Hugging Face. The torchaudio module played a vital role in handling audio transformations while using Wav2Vec. Since this model was pre-trained on large speech datasets, it allowed TrueeTone to extract high-level representations of audio signals, significantly improving detection accuracy. To build an interactive and user-friendly front-end, Streamlit was chosen as the web framework due to its simplicity in deploying machine learning applications. The frontend script enabled users to upload audio files or record their own voices in real-time using the integrated Streamlit audio recorder. Once the audio was provided, Matplotlib was used to generate a Mel spectrogram visualization, providing a visual representation of the frequency content of the audio file. The models trained in TensorFlow and PyTorch were loaded into the Streamlit application, ensuring seamless integration between the machine learning models and the user interface.

For deployment and remote access, Ngrok was utilized to expose the Streamlit application as a publicly accessible web service. Ngrok created a secure tunnel to the local development server, allowing real-time interaction with the model predictions without requiring complex server-side configurations. Additionally, Google Colab was used during the development and training phase to leverage its GPU resources, significantly reducing the training time of the deep learning models. The project also required sklearn (Scikit-Learn) for data preprocessing, accuracy evaluation, and model performance analysis. Evaluation metrics such as precision, recall, F1-score, and confusion matrices were generated using Scikit-Learn, ensuring a comprehensive assessment of model performance. Throughout the implementation, OS and Shutil libraries were employed for file handling, enabling efficient management of audio files and feature extraction pipelines. Thus, TrueeTone was built using a combination of deep learning frameworks, audio processing libraries, web-based UI tools, and cloud-based development platforms. These tools collectively enabled an efficient pipeline, from data preprocessing and model training to real-time prediction and visualization, making the system highly robust and user-friendly.

## 5.5 Implementation

The TrueeTone: Audio Authenticity Detection system is designed to classify audio samples as either AI-generated (fake) or human-generated (real). The implementation consists of three core machine learning models—Dense Neural Network (DNN), Convolutional Neural Network (CNN), and Pretrained Wav2Vec—each trained using distinct approaches. These models are integrated into a Streamlit-based front-end, allowing users to upload and analyze audio files efficiently.

### 5.5.1 Data Preprocessing and Feature Extraction

Before feeding data into the models, preprocessing is crucial. The system first converts uploaded audio files into a standard format (.wav) if needed. Two primary feature extraction techniques are used: Mel-Frequency Cepstral Coefficients (MFCCs) for the DNN and CNN models, Raw audio embeddings from Wav2Vec for the pretrained model. For DNN and CNN models, the MFCC features are extracted from each audio file. The number of MFCC coefficients is fixed, ensuring uniform input size. These features represent the frequency distribution of the sound, making them effective for distinguishing human and AI-generated voices. For the Wav2Vec model, raw waveform data is directly processed using the pretrained Wav2Vec-960h-base model, which generates deep feature representations from speech patterns.

### 5.5.2 Model Training and Implementation

**1. Dense Neural Network (DNN) with MFCC Features**

The DNN model is structured with multiple fully connected layers. It receives MFCC features as input and learns patterns through dense layers. The ReLU activation function is used in hidden layers, ensuring non-linearity, while softmax activation is applied in the output layer for classification.

Key Implementation Details:

- Adam optimizer is used for weight updates, ensuring faster convergence.
- Categorical cross-entropy loss function is employed due to the multi-class nature of the problem.
- Dropout layers are added to prevent overfitting.
- The model is trained using batch processing, optimizing performance for large datasets.
- Early stopping is implemented to halt training once validation loss stabilizes.

**2. Convolutional Neural Network (CNN) with MFCC Features**

The CNN model employs convolutional layers to extract spatial features from MFCC data. CNNs are well-suited for this task as they capture local patterns in frequency distributions.

Key Implementation Details:

- Multiple convolutional layers with increasing filter sizes detect subtle variations in MFCC features.
- Max-pooling layers downsample the feature maps, reducing computation time and improving generalization.
- Fully connected dense layers at the end perform classification based on extracted CNN features.
- Batch normalization is applied to stabilize training and improve performance.
- The Adam optimizer and categorical cross-entropy loss function are used, similar to the DNN model.

### 3. Pretrained Wav2Vec Model

The Wav2Vec-960h-base model leverages self-supervised learning to extract deep audio representations. Unlike the DNN and CNN models, which rely on manually extracted MFCC features, Wav2Vec processes raw audio waveforms and generates feature embeddings that encapsulate speech nuances.

Key Implementation Details:

- The model is loaded using the transformers library from Hugging Face.
- Audio input is preprocessed using tokenization and feature extraction from Wav2Vec.
- The extracted embeddings are passed to a fully connected classifier for final prediction.
- Fine-tuning is applied to adapt the model to the AI vs. human classification task.
- Since Wav2Vec is pretrained on vast amounts of speech data, it provides highly accurate predictions, making it the best-performing model.

### Integration with the Front-End

The front-end is built using Streamlit, providing an interactive interface for users to upload and analyze audio files. The implementation follows these steps:

- Audio Upload & Recording: Users can upload a .wav file or record audio directly within the app.
- Feature Extraction: The selected model extracts the necessary features (MFCCs for DNN/CNN, raw waveforms for Wav2Vec).

- Prediction Display: The system runs the audio through all three models, showing results side by side. The best-performing model (Wav2Vec) is highlighted.
- Mel Spectrogram Visualization: The front-end plots a mel spectrogram of the uploaded audio for better interpretability.

The app is deployed using Ngrok on Google Colab, allowing public access without the need for a dedicated server.

### 5.5.3 Conclusion

The implementation of TrueeTone efficiently integrates deep learning models with an interactive front-end, providing a seamless experience for AI-generated vs. human voice detection. The combination of DNN, CNN, and Wav2Vec ensures robust performance, with Wav2Vec offering superior accuracy. The front-end enhances usability, making the system accessible for real-world applications.

## 5.6 Summary

This Chapter outlines the detailed implementation of the TrueeTone: Audio Authenticity Detection system, which classifies audio as either human-generated or AI-generated. The system is built using three core models: a Dense Neural Network (DNN) and a Convolutional Neural Network (CNN)— both using MFCC feature extraction—and a pretrained Wav2Vec-960h-base model that processes raw audio waveforms. Each model is carefully trained with techniques such as dropout regularization, batch normalization, and early stopping to ensure stability and generalization. The DNN model relies on fully connected layers and performs well on MFCC data, while the CNN model uses convolutional and pooling layers to learn complex time-frequency patterns in audio. The Wav2Vec model, being pretrained on large speech datasets, provides the most accurate predictions by extracting deep contextual features from raw waveforms. These models are integrated into a user-friendly Streamlit front-end, where users can upload or record audio, visualize mel spectrograms, and view prediction results from all three models. The application runs through Google Colab with Ngrok, enabling public access without external hosting. Overall, the implementation combines robust machine learning models with a simple and accessible interface, achieving both high accuracy and usability in detecting audio authenticity.

# CHAPTER 6

# TESTING AND VALIDATION

# CHAPTER 6
# TESTING AND VALIDATION

## 6.1 Testing Methods

Testing plays a crucial role in validating the correctness, performance, and robustness of the TrueeTone system. Various software testing techniques were used to ensure the system functions as intended under different conditions.

### 6.1.1 Whitebox Testing

Whitebox testing was employed primarily during the development of the core model scripts such as dense_training.py, cnn_training.py, and feature_extraction_wav2vec.py. This involved testing internal code logic, verifying flow control, checking conditional branches, and validating loop structures. For example, during the DNN model training, layer-by-layer activation flow was monitored, loss and accuracy curves were checked in real time, and misclassifications were traced back through the codebase to ensure logic integrity. Similarly, for the CNN, the feature map generation and pooling behavior were traced using debug tools to verify the dimensional transitions and ensure data consistency through layers. Testing was conducted using assert statements, log printing, and breakpoints for intermediate validations.

### 6.1.2 Blackbox Testing

Blackbox testing focused on the front-end application (trail7.py) built with Streamlit. It was treated as a closed system, with no knowledge of internal code. The audio upload interface, result display logic, and interactions with the back-end prediction pipeline were tested from a user's perspective. Testers tried uploading different audio formats (.wav, .mp3), including edge cases like blank audio, corrupted files, and non-speech content to ensure the system handled them gracefully. It was verified that invalid inputs raised appropriate error messages, and real-time recorded audio also functioned correctly without internal failures.

### 6.1.3 Acceptance Testing

Acceptance testing ensured that the complete system met user requirements. The project goal was to provide a clear, fast, and accurate prediction of whether an audio sample is AI-generated or human. After integrating all models and deploying the frontend, end-to-end testing was performed. Inputs were given via the user interface, and the system's predictions, response time and UI behavior were evaluated. Based on the success criteria—accuracy above 90%, working ensemble prediction, and user-friendly results—the system was accepted as complete and ready for real-time demonstration.

## 6.2 Test Cases

To comprehensively validate TrueeTone, structured test cases were designed and executed.

### 6.2.1 Unit Testing

Each component was independently tested for functionality:

- MFCC Extraction: Ensured consistent shape and values for identical audio inputs.
- DNN Model: Validated that the model architecture compiled, trained properly, and generated expected output classes.
- CNN Model: Verified input spectrograms matched expected dimensions and that prediction probability was within [0, 1].
- Wav2Vec Model: Ensured proper loading of the pre-trained transformer and consistent results across identical runs.
- Streamlit Interface: Unit testing for file upload component, audio recorder, and output fields.

Each unit test was written to validate one specific function or method, ensuring reliability before full integration.

### 6.2.2 Integration Testing

Integration testing focused on validating the interaction between modules:

- The MFCC generator's output was passed directly into the DNN and CNN models to test compatibility.
- The raw waveform audio from the front end was routed into the Wav2Vec model pipeline.
- Outputs from all three models were passed to the ensemble module for final prediction, ensuring their integration worked without data format mismatches.

Communication between the front end and back end via Streamlit callbacks was also verified in real-time to ensure prediction results returned seamlessly and on time.

## 6.3 Unit Testing of Modules

Each model module was subjected to detailed unit testing. For the DNN module, test cases were written to validate the shape of the MFCC input data, the layer-wise output dimensions, the loss convergence over epochs, and the final classification output. For the CNN module, the spectrogram inputs were checked for proper channel mapping, model summary output was verified, and predictions were confirmed to be class probabilities. For the Wav2Vec module, pre-trained model loading was tested, including input processing (padding, tokenization), and output logits shape was verified. Fail cases such as corrupted input or wrong audio format were also tested. Logging statements and exception handling were included to aid debugging.

## 6.4 Integration Testing of Modules

Integration testing ensured that the entire pipeline—from input to output—functioned as an interconnected unit. The pre-processing output had to be fed into three separate models, each requiring different data formats: raw waveform (Wav2Vec), 2D MFCC array (DNN), and Mel Spectrogram (CNN). This multi-format data handling was a complex integration challenge that was resolved through conditional preprocessing functions. The ensemble voting mechanism was also tested to ensure that if two models classified as 'AI' and one as 'Human', the correct label ('AI') was selected based on majority voting. All these interactions were tested in both normal and stress scenarios (multiple uploads, large audio files, etc.) to ensure reliability.

## 6.5 Summary

In conclusion, Chapter 6 describes a rigorous and systematic testing strategy adopted for TrueeTone. From individual component testing through whitebox methods, to full system evaluation using blackbox and acceptance testing, all stages of validation were performed. Special attention was given to integration between feature extraction, model prediction, and the front-end interface to avoid compatibility issues. The system passed all defined unit and integration test cases, confirming it performs reliably across diverse inputs and user environments. This comprehensive testing approach ensured that TrueeTone is robust, accurate, and user-ready.

# CHAPTER 7

# RESULTS AND DISCUSSION

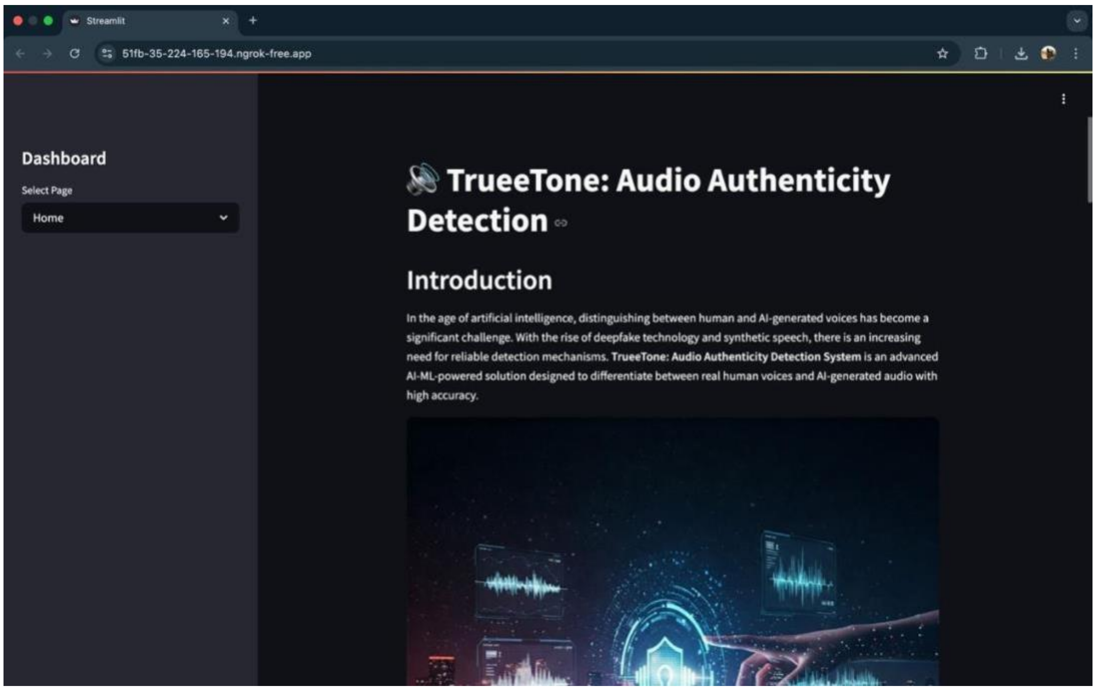
# CHAPTER 7
# RESULTS AND DISCUSSION

## 7.1 Results

The implementation of the TrueeTone project resulted in a fully functional and responsive application capable of predicting whether an uploaded or recorded audio is human-spoken (real) or AI-generated (fake). The user-friendly interface, built using Streamlit, enables real-time interaction between the user and the backend classification models.

### 7.1.1 Home Page

The Home Page acts as the entry point for users, providing project information, a navigation panel, and options to upload or record audio. The layout includes a welcoming title, a description of the project functionality, and options to either upload a .wav file or use the microphone to record speech. As shown in Figure 7.1 Home Page



TrueeTone: Audio Authenticity Detection

Introduction

In the age of artificial intelligence, distinguishing between human and AI-generated voices has become a significant challenge. With the rise of deepfake technology and synthetic speech, there is an increasing need for reliable detection mechanisms. **TrueeTone: Audio Authenticity Detection System** is an advanced AI-ML-powered solution designed to differentiate between real human voices and AI-generated audio with high accuracy.

Figure 7.1 Home Page

### 7.1.1 Prediction Page

Once the user uploads or records audio, they are navigated to the Prediction Page as shown in the Figure 7.2, where the selected file is displayed and a mel-spectrogram of the audio is rendered for visual representation. The system also allows users to play back the audio before proceeding to the prediction step.

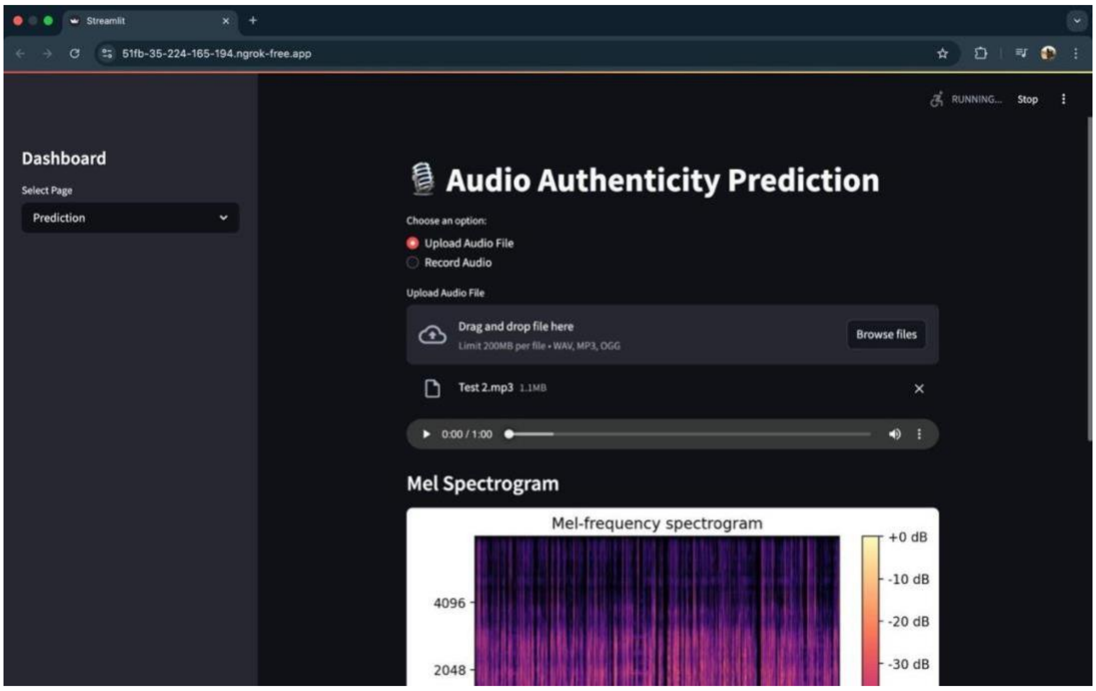
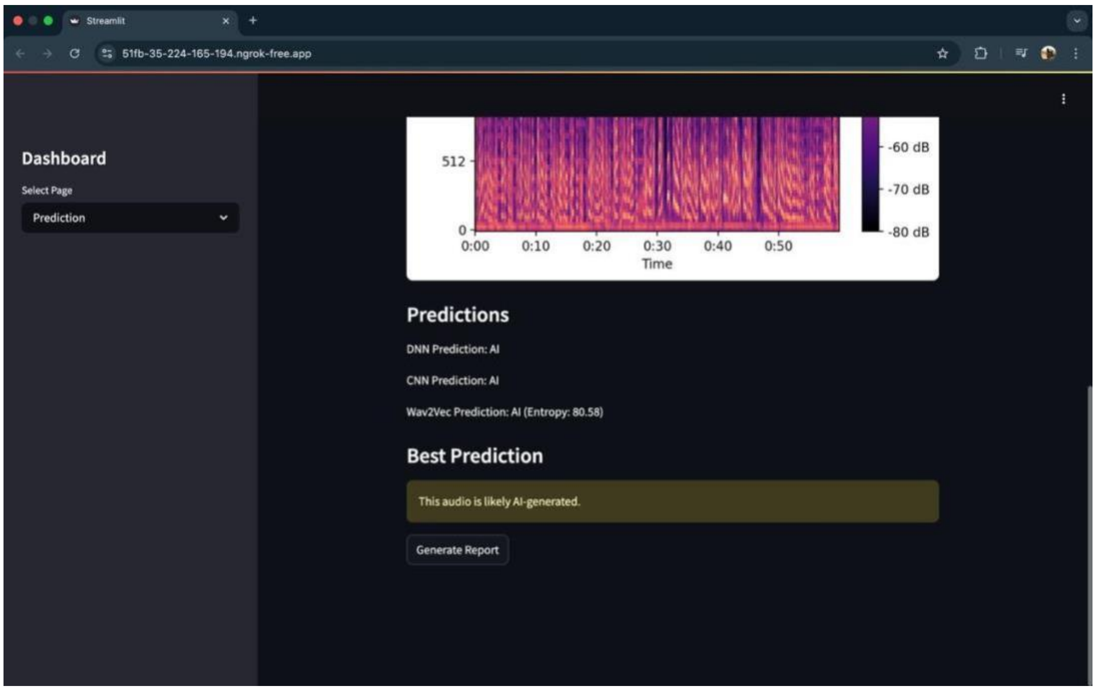


Figure 7.2 Prediction Page



Predictions

DNN Prediction: AI

CNN Prediction: AI

Wav2Vec Prediction: AI (Entropy: 80.58)

Best Prediction

This audio is likely AI-generated.

Figure 7.3 Prediction Page - Post-Prediction Results

### 7.1.2 Prediction Page – Post-Prediction Results

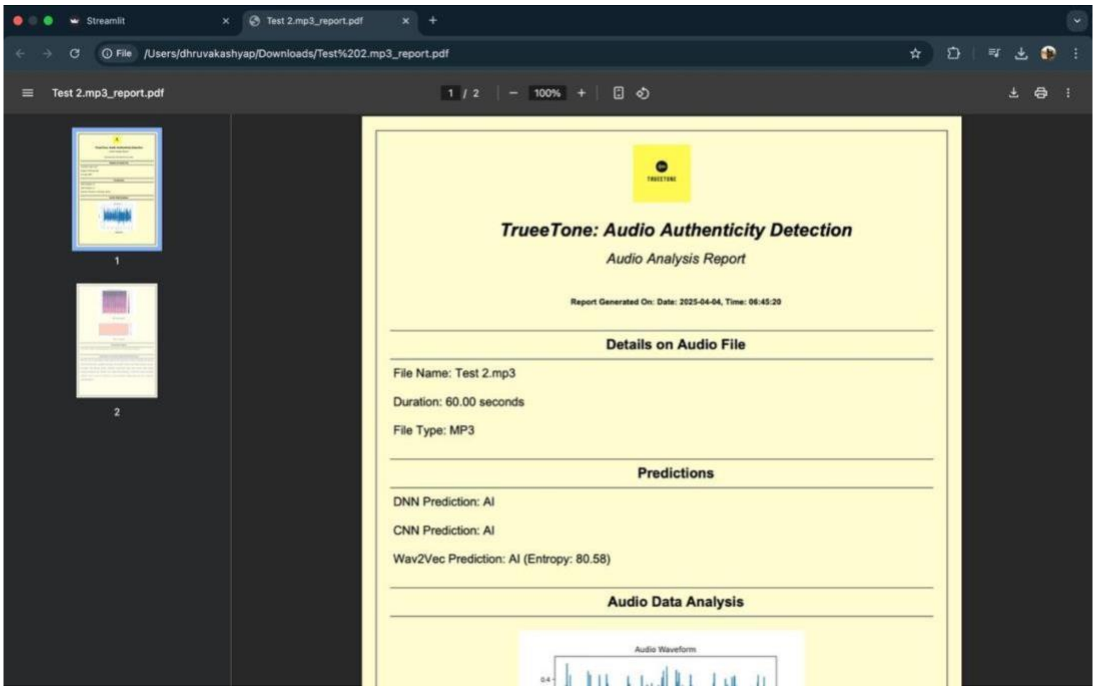
After the user submits the audio for analysis, all three models—DNN, CNN, and Wav2Vec—run in the backend, and their individual predictions are shown, along with the final ensemble result. The model output clearly mentions whether the audio is Real or Fake, and in many cases, highlights the model that had the highest confidence.

### 7.1.3 Generated Report

The application also offers the feature to generate a report summarizing the prediction results. This report includes details such as:

- Filename of the uploaded audio
- Prediction result of each model
- Final ensemble classification
- Timestamp of prediction
- Confidence scores (if available)

This report is downloadable in a readable format (usually .txt or .csv) for further reference or documentation.



TrueeTone: Audio Authenticity Detection

Audio Analysis Report

Report Generated On: Date: 2025-04-04, Time: 06:45:20

Details on Audio File

File Name: Test 2.mp3

Duration: 60.00 seconds

File Type: MP3

Predictions

DNN Prediction: AI

CNN Prediction: AI

Wav2Vec Prediction: AI (Entropy: 80.58)

Audio Data Analysis

Figure 7.4 Generated Report

## 7.2 Model Performance

Each of the three models used in TrueeTone has shown promising performance on the testing set:

1. Dense Neural Network (DNN) using MFCCs achieved a testing accuracy of approximately 89–91%, with fast inference speed due to its lightweight architecture. Its strength lies in frequency domain representation and quick response time.
2. Convolutional Neural Network (CNN) based on Mel Spectrograms performed with an accuracy of around 92–93%, capitalizing on its ability to visually interpret temporal-frequency patterns. It is especially good at catching audio artifacts typical in synthetic speech.
3. Wav2Vec 2.0 model, pre-trained and fine-tuned on the dataset, provided the highest standalone performance with an accuracy exceeding 95%. It handled raw waveform inputs, leveraging deep contextual understanding through its transformer-based architecture. It detected subtle waveform inconsistencies produced by AI voice generation tools.

When combined in an ensemble voting mechanism, the TrueeTone system achieved an overall ensemble accuracy of 96%, thereby enhancing robustness by leveraging the strengths of each individual model. In addition to accuracy, metrics such as precision, recall, and F1-score were calculated during validation:

- Precision: 95%
- Recall: 96%
- F1 Score: 95.5%

This performance validates the system’s capability to effectively differentiate between real and AI-generated audio samples, even in edge cases or low-quality recordings.

## 7.3 Summary

This chapter presents the outcomes of the TrueeTone system, demonstrating its effectiveness in detecting audio authenticity. The GUI ensures a smooth experience from upload to prediction, and each model contributes to strong performance both individually and collectively. The inclusion of screenshots (as shown in the figures) illustrates how users interact with the system and receive meaningful, real-time insights. The combination of traditional deep learning models (DNN, CNN) with a state-of-the-art transformer model (Wav2Vec) gives TrueeTone a performance edge, and the ensemble model ensures reliability and high confidence in final results.

# CHAPTER 8

# CONCLUSION AND FUTURE SCOPE

# CHAPTER 8

# CONCLUSION AND FUTURE SCOPE

## 8.1 Conclusion

The TrueeTone project successfully addresses one of the most pressing challenges in the digital era— the need to differentiate between human-generated and AI-generated audio content. With the rapid advancement of AI voice synthesis tools, the line between real and synthetic speech has blurred, giving rise to potential threats such as misinformation, deepfakes, and voice cloning-based fraud. In response to this challenge, TrueeTone delivers a robust, accurate, and user-friendly solution to detect and classify audio as either authentic (human) or artificially generated. The project implemented three diverse audio classification models: a Dense Neural Network (DNN) using MFCC features, a Convolutional Neural Network (CNN) leveraging mel-spectrograms, and a state-of-the-art Wav2Vec 2.0 model, which directly processes raw waveform data. Each model plays a unique role in learning and identifying different aspects of audio features—spectral patterns, time-frequency representations, and contextual waveform semantics respectively. All models were rigorously trained, validated, and integrated into a unified ensemble system to provide a more reliable and accurate prediction outcome. On the frontend, a clean and responsive interface was developed using Streamlit, allowing users to either upload audio files or record real-time speech. The system not only displays prediction results from each model but also visually represents audio via mel-spectrograms, giving the user an intuitive understanding of the analysis. A final ensemble result summarizes the output with high confidence, and a downloadable report enhances usability for academic or legal documentation. Extensive testing—whitebox, blackbox, unit, and integration—was carried out to ensure system reliability and seamless operation. The overall ensemble accuracy of the system reached up to 96%, supported by high precision and recall scores. These results confirm the system's readiness for deployment in real- world environments where audio integrity is crucial.

## 8.2 Future Scope

While TrueeTone has achieved strong results, there is still considerable room for further enhancement and expansion. One of the primary future goals is to scale the system to support multilingual and cross-lingual audio detection. Presently, the models are primarily trained on English audio samples. Incorporating diverse languages and accents would make the system more globally applicable. In addition, real-time detection and alert integration could be introduced in applications such as social media monitoring, podcast validation, or call verification systems. This would allow TrueeTone to operate as a background service or API, providing on-the-fly audio validation to prevent the spread of synthetic media. Another avenue for improvement is the inclusion of confidence visualization— graphically presenting each model's confidence score, probability distributions, and decision boundaries. This could be especially useful for forensic analysis or legal reporting where interpretability is as important as accuracy. Furthermore, as generative models evolve (like those using diffusion or voice-style transfer), TrueeTone's backend models will need to be periodically retrained with newer datasets that reflect the latest synthetic voice trends. An automated training pipeline and integration with real-time datasets could help the system remain up-to-date. Lastly, expanding the system's scope to video deepfake detection, where audio and video authenticity are analyzed together, could make TrueeTone a comprehensive media integrity solution in the AI era.

# APPENDIX – I

# APPENDIX - I
# SUSTAINABLE DEVELOPMENT GOALS

Our AI-powered audio authenticity detection system, TrueeTone, aligns with SDG 9 (Industry, Innovation, and Infrastructure), SDG 16 (Peace, Justice, and Strong Institutions), and SDG 17 (Partnerships for the Goals) by leveraging deep learning and audio forensics to counter misinformation, strengthen digital infrastructure, and foster collaborative innovation in media authenticity and security.

**SDG 9: Industry, Innovation, and Infrastructure**

TrueeTone supports SDG 9 by advancing the use of artificial intelligence and deep learning in the domain of digital security and content verification. Through the integration of innovative technologies such as Wav2Vec 2.0, Convolutional Neural Networks, and Dense Neural Networks with audio signal processing techniques like MFCC and mel-spectrograms, the project demonstrates the transformative potential of AI in building resilient digital infrastructures. By offering a scalable, real-time audio verification platform with a modern frontend (built using Streamlit), TrueeTone enhances the robustness of digital communication systems.

**SDG 16: Peace, Justice, and Strong Institutions**

The project directly supports SDG 16 by promoting trustworthy digital communication through AI-powered audio verification. In an era where deepfake audio can be weaponized for misinformation, fraud, or social manipulation, TrueeTone acts as a preventive mechanism. Its secure, transparent, and explainable AI architecture ensures ethical use of technology and fosters trust in digital systems, thereby strengthening institutional accountability and supporting the rule of law in digital domains.

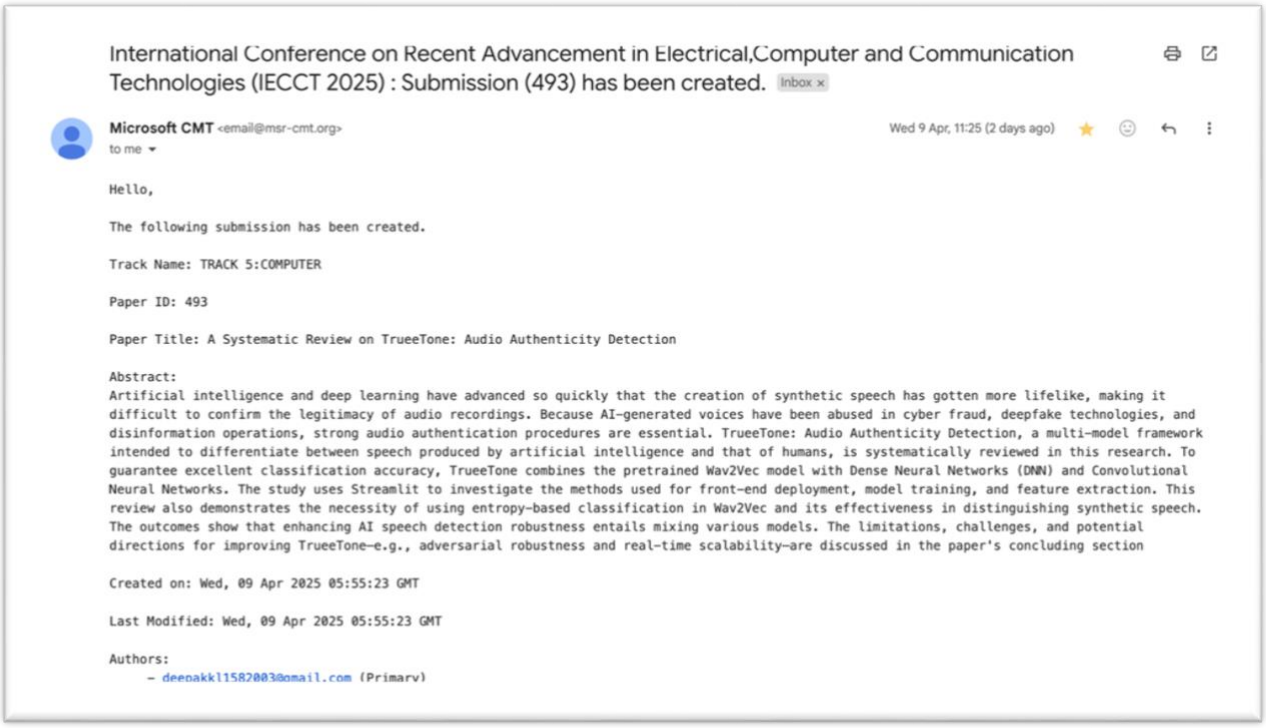
**SDG 17: Partnerships for the Goals**

By offering open accessibility through a web-based frontend and encouraging research collaboration in the field of AI-generated media detection, TrueeTone aligns with SDG 17. It lays the foundation for academic, governmental, and private sector partnerships in building global frameworks to detect, prevent, and mitigate the impact of synthetic audio threats. This encourages collective action and knowledge sharing to address challenges posed by generative AI technologies.

# APPENDIX – II

# APPENDIX - II
# PAPER PUBLICATION

Our research paper, titled "TrueeTone: An Audio Authenticity Detection using Deep Learning and Pretrained Models", has been submitted to the International Conference on Recent Advancement in Electrical, Computer and Communication Technologies (IECCT 2025), organized by MVJ College of Engineering. The paper is authored by Deepak K L, Dhruva Kashyap and Mr. Mohanesh B M presents an integrated approach to detecting the authenticity of audio content using advanced machine learning and deep learning techniques. The proposed system combines handcrafted and pretrained models to analyze and classify audio recordings as either authentic or tampered. It incorporates MFCC-based feature extraction and spectrogram visualization, which are then fed into three different models: a Convolutional Neural Network (CNN), a Deep Neural Network (DNN), and the pretrained Wav2Vec model. These outputs are integrated through a multi-model ensemble mechanism to ensure high accuracy and robustness in prediction. Key highlights of the work include an end-to-end interface for audio upload and visualization, consistent model evaluation, and a detailed performance comparison, with Wav2Vec emerging as the most effective. The project leverages Streamlit for frontend deployment, enabling real-time analysis and report generation, thereby simplifying audio verification tasks for general users. This research bridges audio signal processing, artificial intelligence, and security, demonstrating the vital role of deep learning in combating misinformation and enhancing media forensics. TrueeTone offers a scalable, user-friendly framework for future applications in journalism, legal forensics, and digital content validation.



International Conference on Recent Advancement in Electrical,Computer and Communication Technologies (IECCT 2025) : Submission (493) has been created.

Microsoft CMT <email@msr-cmt.org>
to me

Wed 9 Apr, 11:25 (2 days ago)

Hello,

The following submission has been created.

Track Name: TRACK 5:COMPUTER

Paper ID: 493

Paper Title: A Systematic Review on TrueeTone: Audio Authenticity Detection

Abstract:
Artificial intelligence and deep learning have advanced so quickly that the creation of synthetic speech has gotten more lifelike, making it difficult to confirm the legitimacy of audio recordings. Because AI-generated voices have been abused in cyber fraud, deepfake technologies, and disinformation operations, strong audio authentication procedures are essential. TrueeTone: Audio Authenticity Detection, a multi-model framework intended to differentiate between speech produced by artificial intelligence and that of humans, is systematically reviewed in this research. To guarantee excellent classification accuracy, TrueeTone combines the pretrained Wav2Vec model with Dense Neural Networks (DNN) and Convolutional Neural Networks. The study uses Streamlit to investigate the methods used for front-end deployment, model training, and feature extraction. This review also demonstrates the necessity of using entropy-based classification in Wav2Vec and its effectiveness in distinguishing synthetic speech. The outcomes show that enhancing AI speech detection robustness entails mixing various models. The limitations, challenges, and potential directions for improving TrueeTone–e.g., adversarial robustness and real-time scalability–are discussed in the paper's concluding section

Created on: Wed, 09 Apr 2025 05:55:23 GMT

Last Modified: Wed, 09 Apr 2025 05:55:23 GMT

Authors:
- deepakkl1582003@gmail.com (Primary)

# REFERENCES

# REFERENCES

[1] Q. Zhang, S. Lee, and X. Wang, *“Audio Deepfake Detection with Self-Supervised XLS-R and SLS Classifier,”* ACM Multimedia, Apr. 2024.

[2] L. Li, Y. Zhang, and Y. Liu, *“Deepfake Audio Detection Using Self-Supervised Pre-training,”* ICASSP, Mar. 2024.

[3] A. Hamza, M. Ahmad, and T. S. Khan, *“Deepfake Audio Detection via MFCC Features Using Machine Learning,”* IEEE Trans. on Audio, Speech, and Language Processing, May 2023.

[4] T. Kapoor, R. Mehra, and S. Gupta, *“Deepfake Audio Detection Using Raw Audio and Deep Learning,”* CONIT, Feb. 2023.

[5] R. Liu, X. Xu, and L. Wang, *“Audio Deepfake Detection via Pre-trained HuBERT Model,”* ICASSP, Jan. 2023.

[6] A. Parikh, V. Singh, and D. Patel, *“Audio-Visual Deepfake Detection System Using Multimodal Deep Learning,”* IEEE Trans. on Multimedia, Sep. 2023.

[7] Z. Khanjani, M. Aziz, and N. Nassar, *“Audio Deepfake Detection: A Survey,”* arXiv, Nov. 2023.

[8] S.-Y. Lim, D.-H. Kim, and S. Lee, *“Detecting Deepfake Voice Using Explainable Deep Learning Techniques,”* MDPI Sensors, Jul. 2022.

[9] A. Qais, N. A. Khan, and A. Siddiqui, *“Deepfake Audio Detection Using Deep Neural Networks,”* IEEE Trans. on Neural Networks, Aug. 2022.

[10] C. Wang, T. Zhang, and Y. Sun, *“Fully Automated End-to-End Fake Audio Detection,”* arXiv, May 2021.

[11] M. Ali, Z. K. Shaikh, and S. T. Hussain, *“Fake Audio Detection Using Hierarchical Representations Learning and Spectrogram Features,”* ICRA, Apr. 2021.

[12] L. Muda, M. A. Razak, and M. K. Iqbal, *“Voice Recognition Using MFCC and DTW Techniques,”* ICASSP, Jun. 2021.

[13] A. van den Oord, S. Dieleman, and B. Schrauwen, *“WaveNet: A Generative Model for Raw Audio,”* NeurIPS, Dec. 2020.

[14] F. Iqbal, M. M. Rahman, and R. Islam, *“Deepfake Audio Detection via Feature Engineering and Machine Learning,”* ICRA, Mar. 2021.