EchoShield: Detecting AI-Generated Voices in the Modern Soundscape

By

Shahriyar Hasib (011212085)

Md Mehedi Alam Nahi (011212140)

Faisal Ahmed (011212079)

Md. Sahabuddin Shakil (011213039)

Submitted in partial fulfillment of the requirements
of the degree of Bachelor of Science in Computer Science and Engineering

January 26, 2025

Department of Computer Science and Engineering
United International University

Abstract

Deepfake audio technology enables the creation of synthetic voices that closely mimic genuine speech, posing threats to privacy, security, and trust in digital communications. The misuse of fraud, impersonation, and misinformation requires advanced detection solutions. This research presents *EchoShield*, a framework for detecting deepfake audio using advanced machine learning. The system integrates spectrogram analysis, auditory filters, and hybrid models combining CNNs, RNNs, and transformers to capture spectral and temporal anomalies by employing ensemble learning and lightweight designs. Pre-trained models optimize feature extraction and classification, ensuring scalability and adaptability to diverse datasets. *EchoShield* addresses key gaps in existing detection methods, including limited dataset generalization and isolated feature focus. Evaluated through metrics like F1-score, recall, and Equal Error Rate (EER), the framework demonstrates high accuracy and reliability. This research advances audio forensics by providing scalable tools for authenticating digital audio communications.

Acknowledgments

This work would not have been possible without the input and support of many individuals over the last trimester. We express our heartfelt gratitude to everyone who contributed to it in any way.

First and foremost, we are deeply grateful to Almighty **Allah** for His blessings throughout this journey.

We extend our profound thanks to our honorable supervisors, **Dr. Mohammad Nurul Huda**, *Professor and Head of the Department of CSE*, for his invaluable guidance, continuous support and insightful feedback, and **Md. Tarek Hasan**, *Lecturer, Department of CSE*, for his encouragement and constructive advice that shaped this project significantly.

Our sincere gratitude goes to **Dr. Mohammad Shahriar Rahman**, *Professor, Department of CSE, and Director of CITS*, for his expert suggestions and valuable contributions, which enhanced the quality and depth of this research.

Last but not the least, We owe to our family including our parents for their unconditional love and immense emotional support.

Publication List

[Optional] The main contributions of this research are either published or accepted or in preparation in journals and conferences as mentioned in the following list:

Journal Articles

1.

Conference Papers

1.

Additional Publications

Following is the list of relevant publications published in the course of the research that is not included in the thesis:

1.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the digital age, deepfake audio technology has become a disruptive force, making it possible to create artificial voices that remarkably resemble human speech. These developments, fueled by deep learning and artificial intelligence, have found legitimate applications in entertainment, virtual assistants, and content creation. However, because deepfake audio may be used for fraud, impersonation, and the spread of false information, its misuse poses serious risks to privacy, security, and trust.

Deepfake audio detection is a constant struggle since contemporary synthesis methods take advantage of the subtleties of human speech to get beyond conventional detection systems. Early detection attempts frequently lacked the sophistication of more sophisticated deepfake techniques, relying instead on simple feature extraction and traditional machine learning models.

The goal of this research is to overcome these constraints by creating a strong framework for identifying deepfake audio using cutting-edge techniques. This study aims to detect the distinct artifacts found in synthetic speech by utilizing hybrid deep-learning models and collecting important spectral and temporal properties from audio data. The goal is to develop a system that can handle a variety of datasets and synthesis methods while remaining accurate and generalizable.

Assuring that the authenticity of audio data can be consistently confirmed in controlled settings, the project's results are anticipated to improve the security of digital audio content and advance the field of audio forensics.

1.1 Project Overview

1.1.1 Background

Artificial intelligence (AI) is used in deepfake audio technology to create realistic-sounding synthetic voices that can imitate real people. Although these technologies can be used for pleasure, when misused, they can also be extremely dangerous. Conventional detection methods frequently depend on manually created features such as MFCCs and simple machine learning models, which are inadequate in the face of contemporary synthesis methods. Adapting to the subtleties of deepfake audio and guaranteeing reliable performance across a variety of datasets are two current challenges.

The goal of this research is to increase the accuracy of fake detection by utilizing cutting-edge techniques like feature engineering and hybrid deep learning models. In contrast to real-time implementation, the project prioritizes document-based evaluations and dataset-driven experimentation, which is in line with the academic and research-oriented objectives of creating dependable detection systems.

1.1.2 Problem Statement

The rapid advancement of deep learning has enabled the creation of highly realistic deepfake audio, posing significant risks to the authenticity, security, and privacy of digital communications. These synthetic voices, often indistinguishable from genuine speech, are increasingly used for malicious purposes, such as fraud, impersonation, and spreading misinformation. Despite progress in detection techniques, existing methods struggle to generalize across diverse datasets and fail to effectively address the subtle artifacts embedded in audio generated by advanced synthesis methods. This challenge highlights the urgent need for improved solutions to safeguard digital audio integrity.

1.2 Motivation

As deepfake audio technology has grown in popularity, it has become increasingly difficult to verify the legitimacy of digital conversations. There is a serious risk to privacy, security, and public trust when sophisticated AI algorithms are able to produce artificial voices that accurately replicate actual human speech. The financial, media and public safety sectors can all be impacted by the use of these deepfake audios as a weapon for fraud, impersonation and disinformation.

The opportunity to advance machine learning and signal processing methods in order to detect subtle and distinctive artifacts incorporated into synthetic audio provides the computational incentive for solving this issue. Using cutting-edge algorithms that can process high-dimensional, temporal, and spectral characteristics of voice data is necessary

to detect these aberrations.

Resolving this issue has broad advantages. It makes digital verification systems more reliable, protects people and organizations from deepfake-based abuses, fosters confidence in digital ecosystems. Additionally, the study advances the science of audio forensics and strengthens our defenses against malevolent AI applications while promoting safe and moral technological development.

1.3 Objectives

Our research focuses on finding and extracting distinctive features in artificial speech, creating sophisticated machine-learning models, assessing the robustness of various datasets, improving performance for efficiency and accuracy, and advancing the field of audio forensics by providing scalable solutions to prevent the abuse of deepfake audio (see Figure 1.1).

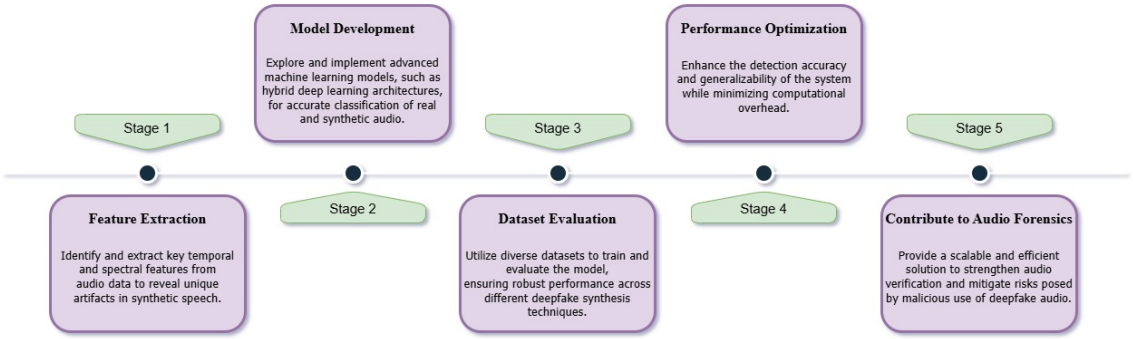

Figure 1.1: Key Objectives for Deepfake Audio Detection Framework

1.4 Methodology

The *EchoShield* methodology is designed to create a robust framework for detecting deepfake audio. The process begins with collecting diverse datasets, including ASVspoof, Codecfake and generating synthetic audio. Preprocessing steps such as segmentation, noise reduction, normalization, and data augmentation ensure high-quality input. Feature extraction focuses on three areas: spectrogram-based features to analyze time-frequency patterns, auditory filters to mimic human hearing, and temporal features to capture rhythm and energy.

A hybrid model architecture combines CNNs for spatial analysis, RNNs for sequential pattern detection, and Transformers to capture long-range dependencies. Pre-trained models are leveraged to extract semantic embeddings for better classification. Ensemble learning integrates predictions from these models to improve robustness.

Performance metrics, such as F1-score, recall, accuracy, and precision, are used to assess how effective the suggested framework is. The methodology ensures strong performance across datasets, balancing accuracy and scalability for advanced deepfake detection.

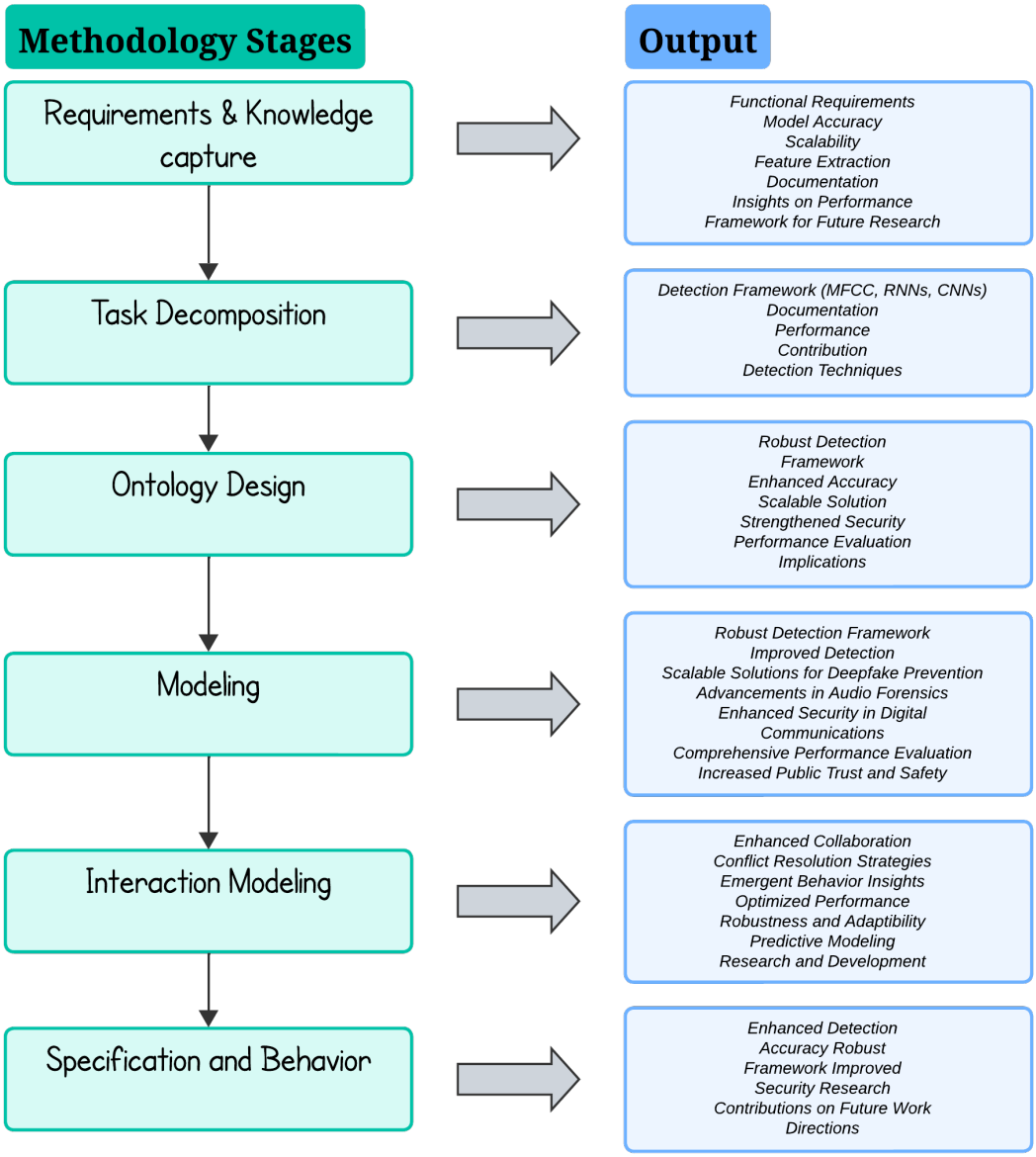

Figure 1.2: Basic Methodology

1.5 Project Outcome

Development of a Robust Deepfake Audio Detection Framework for Enhanced Security, Authenticity, and Forensic Applications. The project outcome mapping is depicted in Figure 1.3.

Figure 1.3: Project Outcome Mapping

1.6 Organization of the Report

This report is structured into six chapters, each addressing critical aspects of the research on detecting deepfake audio. The chapters are organized as follows:

- **Chapter 1: Introduction**
  This chapter introduces the research topic, *"EchoShield: Detecting AI-Generated Voices in the Modern Soundscape,"* highlighting the growing concern over synthetic audio and its impact on privacy, security, and misinformation. The chapter outlines the research problem, objectives, methodology, and structure of the thesis.

- **Chapter 2: Background**
  This chapter provides a review of the evolution of AI-generated audio and existing detection methods. It identifies limitations in current approaches and highlights the research gap that motivates this study.

- **Chapter 3: Methodology**
  The design and framework of the proposed detection system are discussed, detailing feature extraction methods, machine learning models, and system architecture. This chapter includes diagrams to illustrate the workflow and outlines the research plan.

- **Chapter 4: Results and Analysis**
  This chapter presents the outcomes of implementing the *EchoShield* system, including performance evaluations and comparative analysis. Key findings, system accuracy, and areas for improvement are highlighted.

- **Chapter 5: Discussion and Standards**
  This chapter explores the broader implications of the findings, addressing design constraints, ethical considerations, and compliance with relevant standards. A cost analysis and strategies to address system limitations are also included.

- **Chapter 6: Conclusion and Future Work**
  The final chapter summarizes the research contributions, acknowledges study limitations, and suggests directions for future work, such as real-time detection and advanced model integration.

Chapter 2

Background

This chapter provides an overview of the key concepts and technologies related to deepfake audio detection, laying the foundation for the research presented in this report. The focus is on the role of deep learning techniques, specifically Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), in identifying synthetic audio content. In addition, the chapter explores the challenges faced in deepfake audio detection, the limitations of traditional detection methods, and the need for advanced hybrid models that can capture both temporal and spectral features of audio data. The section also reviews existing detection systems and highlights the gap that this research aims to address.

2.1 Preliminaries

This section provides the essential background to understand the methodologies and techniques discussed in the report. The focus is on deep learning models, particularly Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), which are widely used for audio analysis. These models, along with techniques like transfer learning and hybrid architectures, form the foundation for detecting anomalies in deepfake audio. The report also explores the use of pre-trained models for feature extraction, enabling more efficient and accurate detection of manipulated audio content.

2.1.1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are deep learning models specifically designed for processing structured grid data, similar as images or spectrograms. In this exploration, CNNs are employed to:

i. Capture spatial features in spectrogram-based audio representations.

ii. Recognize patterns such as frequency variations and harmonic structures, which are crucial for identifying anomalies in deepfake audio.

iii. Example: *EfficientNet* and *ResNet* architectures offer robust performance by leveraging hierarchical feature extraction and efficient computation.

2.1.2 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are designed for processing sequential data, making them particularly useful for modeling temporal dependencies. In this study, the focus is on the *Bi-LSTM* (Bidirectional Long Short-Term Memory), an RNN variant, which is used to:

i. Analyze temporal coherence and prosodic features like rhythm, stress, and intonation.

ii. Handle long-range dependencies in the audio signals to detect subtle temporal anomalies.

iii. The *Bi-LSTM* ensures that both past and future context is considered in decision-making, improving the model's ability to detect subtle changes in audio over time.

2.1.3 Transfer Learning

Transfer learning is a technique that leverages pre-trained models, which have been trained on large datasets, to adapt to specific tasks. In this project:

i. Pre-trained audio models like *Whisper* and *SpeechBrain* are used to extract embeddings that encapsulate high-level audio features such as pitch, tone, and speaker characteristics.

ii. These embeddings are fine-tuned using a task-specific classifier (e.g., Multi-Layer Perceptron (MLP)) to distinguish real from fake audio samples.

iii. This approach accelerates convergence and enhances generalization on limited datasets by utilizing pre-learned audio representations.

2.1.4 Hybrid Models

Hybrid models combine the strengths of multiple architectures to improve performance. In this research:

i. CNN-RNN Hybrid: This model combines CNN's spatial feature extraction with RNN's temporal modeling to capture both static and dynamic features in spectrograms.

ii. Ensemble Learning: Decision-level fusion of CNN and RNN outputs enhances robustness by leveraging the diverse perspectives offered by both models.

iii. The hybrid approach helps mitigate the weaknesses of individual models, leading to improved overall accuracy.

2.1.5 Pre-Trained Models

Pre-trained models are leveraged to reduce the computational cost of training and to utilize generalized audio representations learned from large datasets. Models such as *Whisper*, *Pyannote*, and *SpeechBrain* provide embeddings that are:

i. Robust to noise and distortions.

ii. Tailored for tasks like speaker recognition, which are highly relevant to detecting anomalies in deepfake audio.

iii. By incorporating pre-trained models, this project aims to enhance the robustness and efficiency of the detection system, ensuring that it remains effective even under various distortions and noise conditions.

2.1.6 Vision Transformer (ViT)

The Vision Transformer (ViT) is a transformer-driven model originally designed for image processing tasks, but it proves highly effective in analyzing spectrograms for audio data. Unlike traditional convolutional neural networks (CNNs), ViT employs self-attention mechanisms to understand global relationships within data, making it particularly well-suited for detecting subtle irregularities in spectrogram-based representations.

ViT plays a crucial role in the following::

i. Leveraging Attention Mechanisms: Effectively capturing long-range dependencies within spectrograms to pinpoint deepfake-related anomalies.

ii. Patch-Based Processing: Dividing spectrograms into smaller patches to extract global features efficiently, ensuring detailed analysis.

iii. Enhancing Model Synergy: Acting as a complementary element to CNNs and RNNs, the attention mechanisms in ViT add depth to feature extraction and decision-making processes.

2.2 Literature Review

The rapid rise of deepfake audio technologies has posed significant challenges for ensuring the authenticity and security of voice-based systems, with existing detection methods often falling short in terms of accuracy, scalability, and robustness. The need for a comprehensive, multi-faceted detection framework has become critical as synthetic audio becomes increasingly sophisticated, exploiting both temporal coherence and spectral intricacies.

Recent studies have made strides in this domain. Techniques like Temporal Deepfake

Location (TDL) leverage Wav2Vec2 embeddings and temporal convolution to detect partially spoofed audio at the frame level, effectively capturing temporal inconsistencies. Similarly, approaches integrating Mean Opinion Scores (MOS), such as MOS-FAD, enhance detection pipelines by focusing on perceptual artifacts and employing novel fusion methods like Gated MLP. Spectrogram-based models exploit transformations such as STFT and Wavelet, paired with hybrid architectures like CNNs and C-RNNs, to detect spectral irregularities, while the Codecfake dataset highlights codec-induced artifacts using LCNN and SSL-based approaches.

Despite these advances, existing frameworks often focus on isolated dimensions-temporal, spectral, or perceptual without integrating them into a unified system. Additionally, scalability to diverse datasets, real-time detection capabilities, and robust generalization remain underexplored. This gap underscores the potential for methodologies that combine adaptive feature extraction, ensemble learning, and multi-modal integration to address the limitations of current approaches and achieve state-of-the-art performance in detecting audio deepfakes.

2.2.1 Similar Applications

Application	**Description**
Resemble Detect [1]	A deepfake voice detection tool using deep neural networks to analyze time-frequency embeddings, detecting subtle inconsistencies in AI-generated audio with over 98% accuracy. Provides robust safeguards against synthetic disinformation.
PlayHT Voice Classifier [2]	Uses machine learning algorithms to distinguish between real and synthetic voices. Offers detailed analysis reports with a user-friendly interface, suitable for both professional and casual use.
Deepware Scanner [3]	An open-source tool combining machine learning and spectral analysis for detecting synthetic audio and audiovisual deepfakes. Supports applications like social media verification and forensic analysis.
Loccus AI [4]	Focuses on fraud prevention by integrating AI voice detection into communication systems. Supports multiple formats and languages.
Reality Defender [5]	Detects audio deepfakes by analyzing synthesis artifacts. Offers solutions for governments, enterprises, and platforms to identify manipulated audio.

Table 2.1: Summary of Similar Applications

2.2.2 Related Research

Title	**Methodology**	**Result**
Deepfake Audio Detection Using Spectrogram-based Feature and Ensemble of Deep Learning Models [6]	∗ Spectrogram-Based Feature Extraction: STFT, LFCC, Wavelet. ∗ End-to-End Models: CNN Baseline, RNN Baseline (Bi-LSTM), C-RNN Baseline. ∗ Transfer Learning: ResNet, ConvNeXt, Swin-Tiny. ∗ Audio Embedding Models: Whisper, SpeechBrain, Pyannote.	EER: 0.03%
The Codecfake Dataset and Countermeasures for the Universally Detection of Deepfake Audio [7]	∗ Feature Extraction: Mel-Spectrograms, Wav2Vec2-XLS-R Embeddings. ∗ Classification: CNN, LCNN. ∗ AASIST: Temporal Modeling techniques (LSTM) + softmax activation.	EER: 0.616%
MOS-FAD: Improving Fake Audio Detection via Automatic Mean Opinion Score Prediction [8]	∗ Train a MOS Predictor. ∗ Feature extraction and detection: Wav2Vec. ∗ Fusion Techniques: Score fusion, embedding fusion. ∗ Gated MLP.	EER: 0.96%
SPECDIFF-GAN: A Spectrally-Shaped Noise Diffusion GAN for Speech and Music Synthesis [9]	∗ Noise Diffusion Preprocessing. ∗ Generator: HiFi-GAN Architecture. ∗ **Discriminators:** Multi-Period and Multi-Resolution Discriminators.	Not mentioned
An Efficient Temporary Deepfake Location Approach Based on Embeddings for Partially Spoofed Audio Detection [10]	∗ Feature Extraction: Wav2Vec2. ∗ Embedding Similarity Module. ∗ Temporal Convolution Operation. ∗ Classification: 1-D CNN.	Not mentioned

A Comprehensive Survey with Critical Analysis for Deepfake Speech Detection [11]	∗ Spectrogram-Based: STFT, Mel-Spectrogram. ∗ Auditory Filters: LFCC, Wavelet. ∗ End-to-End Models: CNN, RNN, C-RNN. ∗ **Ensemble Models:** Multiple Inputs, Multiple Branches. ∗ Transfer Learning: ResNet, MobileNet, EfficientNet. ∗ Audio Embedding Models: XLS-R, Hubert, WavLM, Whisper.	EER: 0.03%
Is Synthetic Voice Detection Research Going into the Right Direction [12]	∗ STFT. ∗ LFCC, Wavelet. ∗ CNN Baseline, RNN Baseline (Bi-LSTM), C-RNN Baseline. ∗ ResNet, ConvNeXt, Swin-Tiny. ∗ Whisper, SpeechBrain, Pyannote.	EER: 0.05%
AntiDeepFake: AI for Deep Fake Speech Recognition [13]	∗ Mel-spectrogram. ∗ Pitch, Shimmer, RMSS, Jitter, Chroma STFT, Spectral Centroids, Spectral Bandwidths, Rolloffs, Zero Crossing Rates, MFCC (Mean, Standard Deviation). ∗ Formants (F1, F2, F3, F4).	Not mentioned
Detecting Deep Fake Voice Using Machine Learning [14]	∗ Spectrogram-based features: STFT, MFCC, Wavelet. ∗ Temporal and frequency domain analysis. ∗ CNN. ∗ Bi-LSTM. ∗ Transfer Learning. ∗ Pre-trained architectures: Whisper, SpeechBrain. ∗ Ensemble Learning	EER: 0.02%
Detecting Deepfake Voice Using Explainable Deep Learning Techniques [15]	∗ STFT, Mel-filterbanks (applied for perceptual emphasis on low-frequency spectra). ∗ CNN. ∗ LSTM. ∗ XAI Methods. ∗ Deep Taylor Decomposition. ∗ Integrated Gradients. ∗ Layer-Wise Relevance Propagation (LRP). ∗ Griffin-Lim Algorithm.	EER: 0.03%

Audio Deep Fake Detection with Sonic Sleuth Model [16]	∗ Audio preprocessing to handle noise. ∗ CNN for feature extraction and classification. ∗ Generalization tested on multiple datasets.	Internal Dataset: Accuracy: 98.27% EER: 0.016 External Dataset: Accuracy: 84.92% EER: 0.085
CROSS-MODALITY AND WITHIN-MODALITY REGULARIZATION FOR AUDIO-VISUAL DEEPFAKE DETECTION [17]	∗ Feature Extraction: Audio and visual features are separately extracted. ∗ Fusion: Audio-visual features are combined using a transformer. ∗ **Cross-modality Regularization:** Aligns audio-visual signals through contrastive learning. ∗ Within-modality Regularization: Refines unimodal features using margin-based and cross-entropy regularization.	Accuracy: 94.1%, EER: 4.3%
WaveFake: A Dataset to Facilitate Audio Deepfake Detection [18]	∗ Dataset Creation: 117,985 audio clips from six architectures in two languages (English and Japanese). ∗ Audio Analysis: Conducted frequency analysis using spectrograms to identify subtle differences. ∗ Classifier Implementation: Signal processing techniques. ∗ Attribution Method: BlurIG for influential audio features. ∗ Evaluation Metrics: Equal Error Rate (EER). ∗ Ethical Considerations.	EER: 0.062%
Detection of Deepfake Environmental Audio [19]	∗ Audio embeddings are pre-trained: VGGish, CLAP, and PANN. ∗ Dense-layer architecture with ReLU activation and dropout. ∗ Binary classification with a sigmoid output. ∗ Binary cross-entropy loss.	Accuracy: 98.02%

Table 2.2: Research Papers Review

2.3 Gap Analysis

The gap analysis for *EchoShield* is visually structured as follows:

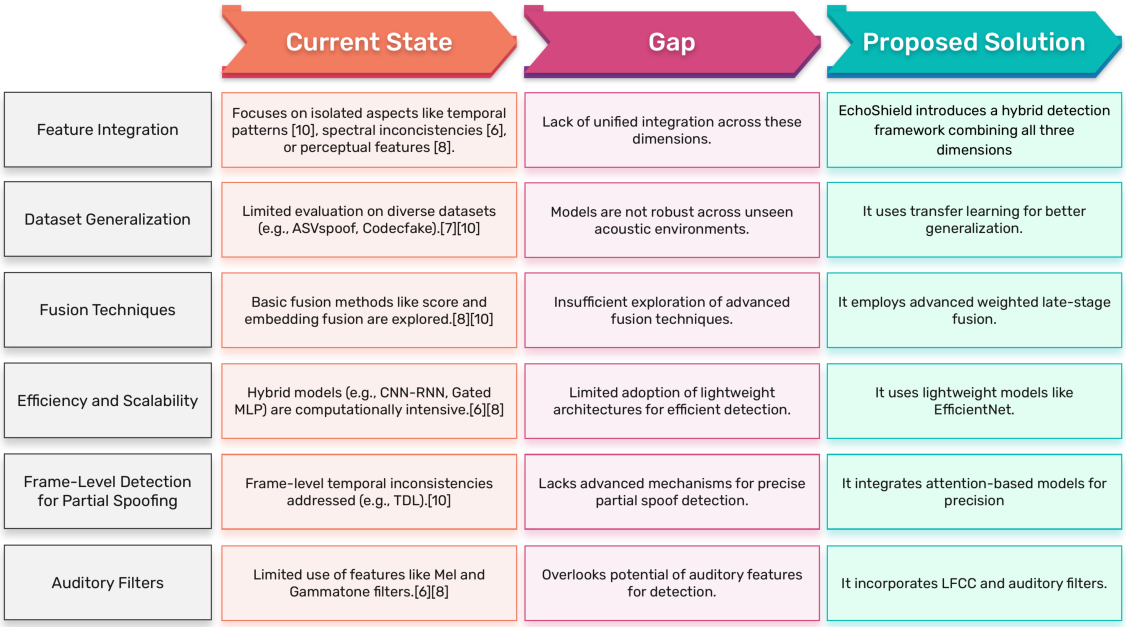

Figure 2.1: Gap Analysis for *EchoShield*

This structured visualization highlights how *EchoShield* can fill gaps and advance audio deepfake detection technology.

2.4 Summary

Chapter 2 focuses on the foundational aspects of deepfake audio detection, emphasizing the role of advanced machine learning techniques like CNNs and RNNs. Key technologies are highlighted, such as transfer learning, hybrid architectures, and pre-trained models, which enhance detection accuracy and generalizability. The literature review reveals gaps in current approaches, including limited scalability and isolated feature focus. The chapter proposes *EchoShield* as a comprehensive solution, combining robust feature extraction, hybrid model integration, and lightweight designs to address these limitations effectively.

Chapter 3

Project Design

This chapter describes the system design, including requirement analysis, techniques and work allocation. It specifies functional and nonfunctional requirements, contextual and data flow diagrams, and user interface design. It also discusses the chosen methodology, alternative solutions, and project strategy to achieve clear alignment with objectives and strong system performance.

3.1 Requirement Analysis

The Requirement Analysis section outlines the functional and non-functional requirements of the *EchoShield* framework, ensuring a clear understanding of the project's goals and necessary features.

3.1.1 Functional and Nonfunctional Requirements

Functional Requirements	**Non-Functional Requirements**
Audio Input Handling	Scalability
Feature Extraction	Accuracy and Robustness
Hybrid Model Architecture	Extensibility
Ensemble Learning	Cross-Dataset Generalization
Classification	Processing Time
Performance Reporting	

Table 3.1: Functional and Non-Functional Requirements

Functional Requirements

1. **Audio Input Handling:** Audio files or segments in standard formats, such as WAV or MP3, must be accepted by the system.
2. **Feature Extraction:** Spectral, temporal and embedding information must be extracted by the system utilizing models like as Whisper and Wav2Vec2 and methods such as STFT, CQT and ZCR.
3. **Hybrid Model Architecture:** Combine Transformer models, CNNs and RNNs to provide reliable feature analysis.
4. **Ensemble Learning:** Integrate results from several models by applying weighted decision fusion and late stage fusion.
5. **Classification:** Classify audio segments as authentic or fraudulent with a 90% accuracy rate or higher.
6. **Performance Reporting:** Produce quantitative measures including F1-score, recall, accuracy and EER.

Non-Functional Requirements

1. **Scalability:** Datasets containing thousands of audio samples must be handled by the system without experiencing performance deterioration.
2. **Accuracy and Robustness:** Reach high detection accuracy (>95%) while reducing false positives in a variety of datasets.
3. **Extensibility:** Encourage integration with cutting-edge models and novel feature extraction strategies.
4. **Cross-Dataset Generalization:** Ensure that performance is consistent across datasets, such as ASVspoof and Codecfake.
5. **Processing Time:** Batch processing for non-real-time applications should be finished within reasonable time constraints.

3.1.2 Context Diagram

The context diagram shows how the *EchoShield* Framework interacts with external entities and data flows, giving a high-level perspective of the system. External entities such as Audio Data Sources and System Administrators/Users communicate with the central system, *EchoShield*. Real and AI-generated raw audio files are supplied to the system by the Audio Data Sources, and the System Administrator/User controls detection requests and results retrieval. Important elements that enable data management and storage, including the Labeled Dataset Repository, Trained Model Repository, and Detection Logs

Repository, guarantee smooth module integration. Primary data exchanges and their relationships with the core framework are depicted in this Figure 3.1.

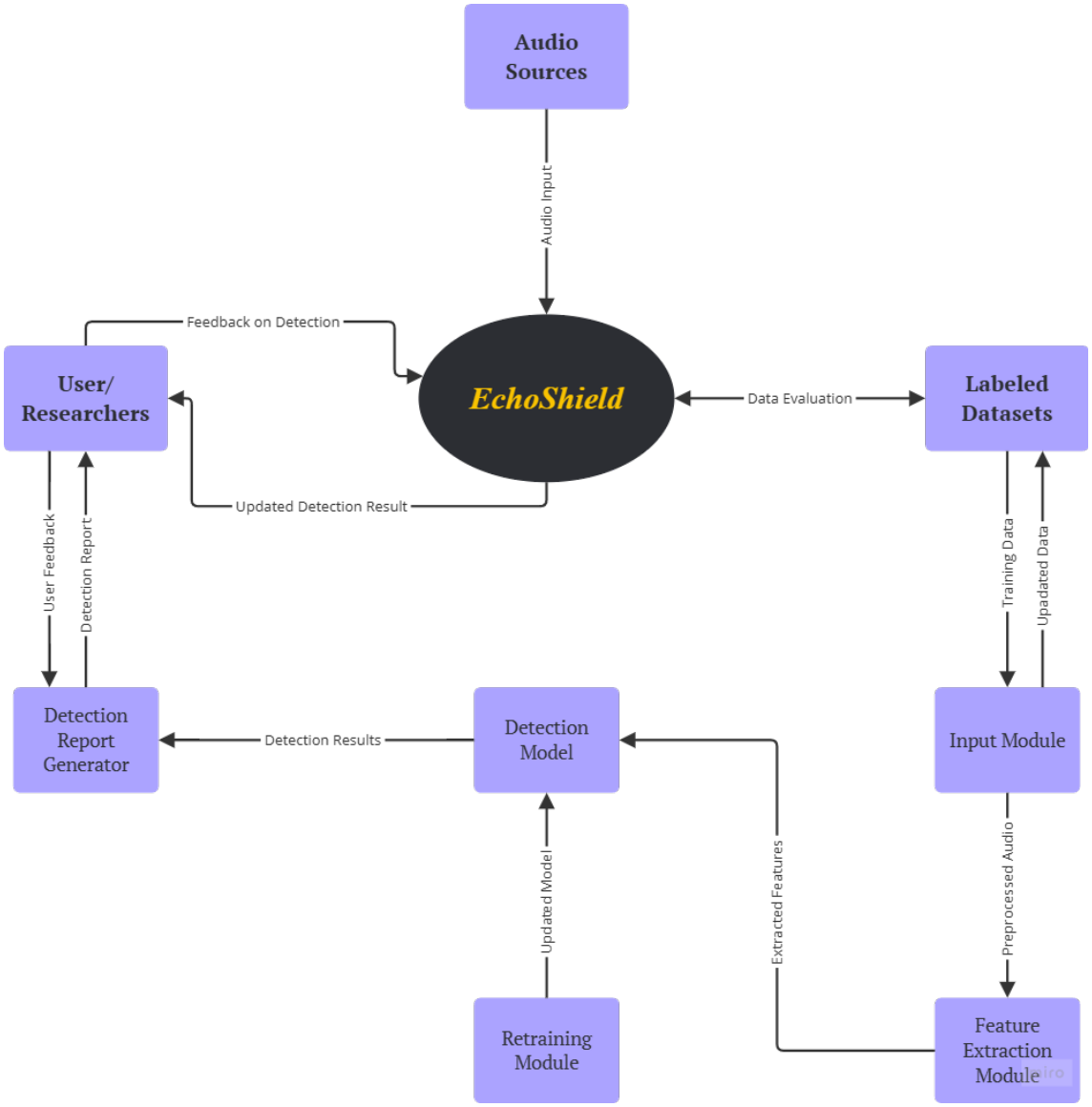

Figure 3.1: Context Diagram

3.1.3 Data Flow Diagram Level 1

The Level 1 Data Flow Diagram shows how data flows between processes, data storage, and external entities while going deeper into the *EchoShield* Framework's functional components. The Labeled Dataset Repository stores the raw audio data after it has been processed and labeled by the Dataset Manager Module. Key audio features are extracted by the Feature Extraction Module for use in detection and model training. The Model Training and Refinement Module builds and updates models kept in the Trained Model Repository using labeled data and extracted features. The Detection System logs the

outcomes of its classification of input audio using trained models in the Detection Logs Repository. This graphic offers a thorough understanding of the EchoShield system's underlying operations and interactions (see Figure 3.2).

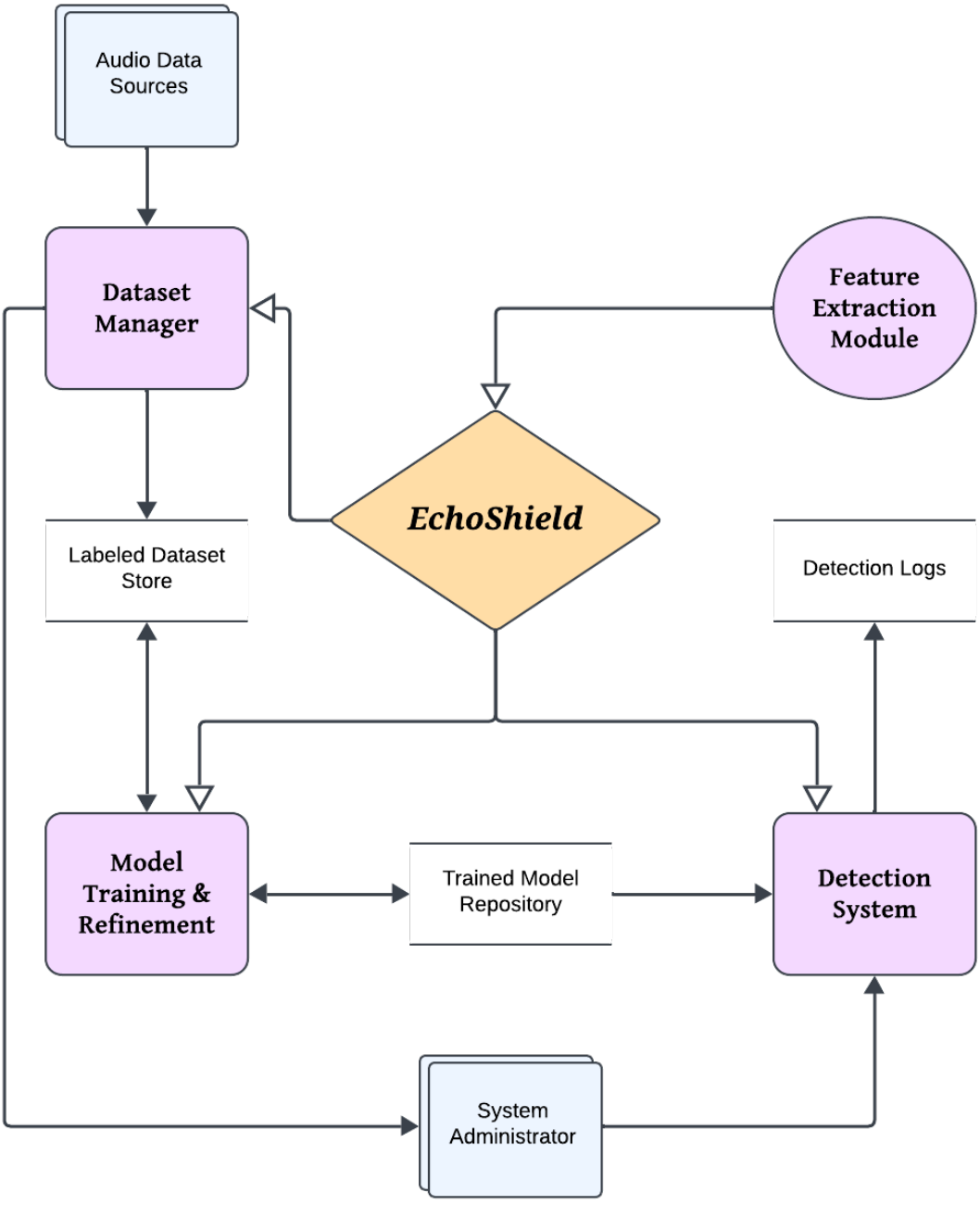

Figure 3.2: Data Flow Diagram Level 1

3.1.4 UI Design

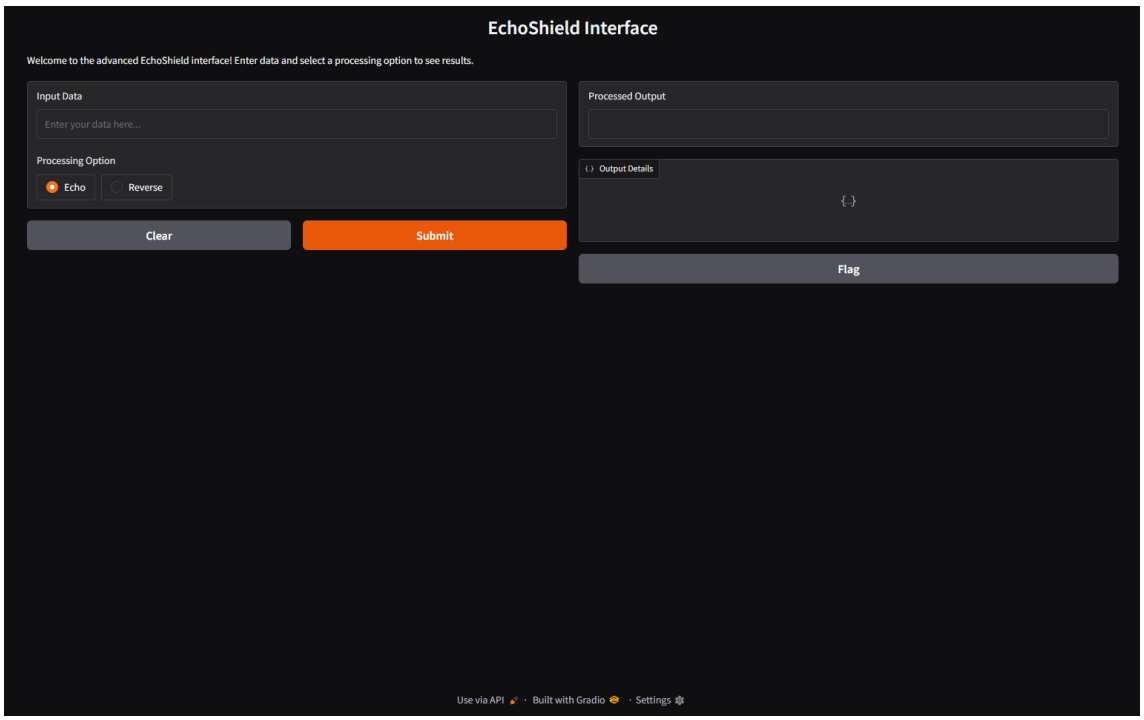

Figure 3.3: Basic Level *EchoShield* Interface

3.2 Detailed Methodology and Design

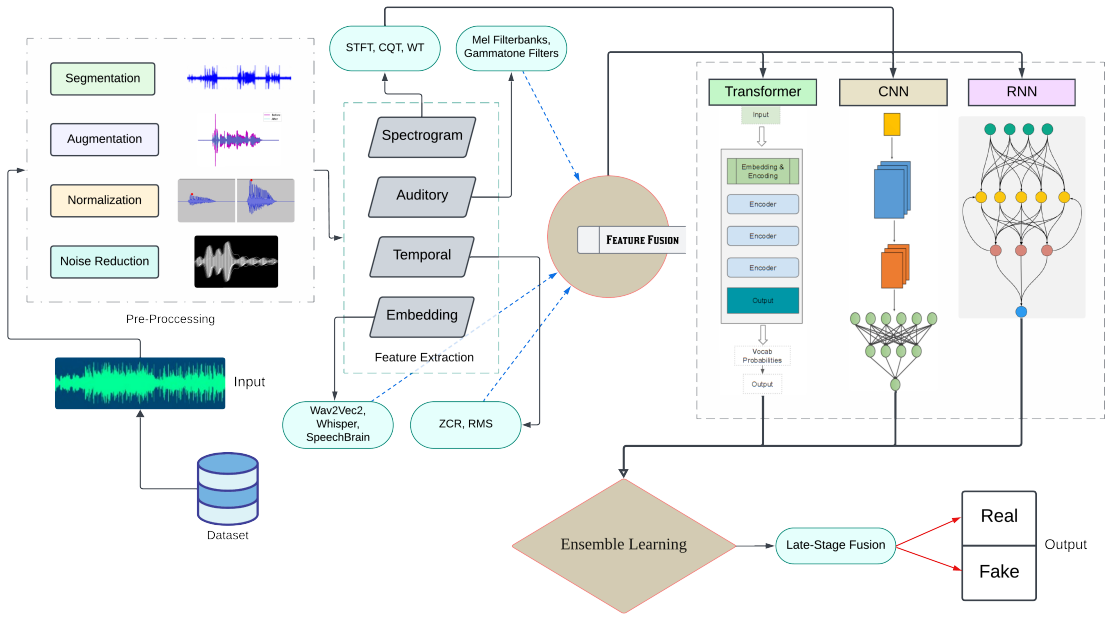

Figure 3.4: Detailed Methodology Architectural Design

1. **Data Acquisition and Preprocessing**
   To ensure the creation of a robust and diverse dataset, a variety of real and synthetic audio samples are collected. Publicly available datasets such as MOS-FAD, ASVspoof, and Codecfake are utilized alongside synthetic data generated using advanced neural audio codecs.

   i. Segmentation: Audio files are divided into fixed durations (e.g., 2–5 seconds) for uniformity.

   ii. Noise Reduction: Techniques like noise filtering and spectral subtraction are applied to eliminate distortions and background noise.

   iii. Normalization: All audio samples are standardized to consistent amplitudes and sampling rates (e.g., 16kHz).

   iv. Augmentation: The dataset is enhanced by introducing controlled variations like noise injection, time stretching, and pitch shifting.

2. **Feature Extraction**
   The aim is to generate meaningful audio representations for effective classification. The extracted features fall into three primary categories:

   Spectrogram Features:

   i. Short-Time Fourier Transform (STFT): Captures time and frequency representations.

   ii. Constant-Q Transform (CQT): Provides logarithmic frequency resolution, ideal for tonal audio.

   iii. Wavelet Transform (WT): Detects localized frequency variations.

   Auditory Filters:

   i. Mel Filterbanks: Mimic the human auditory perception system.

   ii. Gammatone Filters: Enhance speech-specific frequency characteristics.

   Temporal Features:

   i. Zero Crossing Rate (ZCR): Analyze rhythmic and energy-based signal patterns.

3. **Model Architecture**
   A hybrid approach is employed, combining embedding-level, temporal, and spatial analyses.

   i. CNN for Spatial Features: Models like ConvNeXt, ResNet, and EfficientNet focus on detecting spatial anomalies and high-frequency artifacts from spectrograms.

ii. RNN for Temporal Features: Bi-LSTM and GRU are utilized to model sequential dependencies and detect temporal inconsistencies.

iii. Transformer-based Attention Mechanisms: Vision Transformer (ViT) models are used to capture long-range dependencies and contextual patterns.

iv. Pre-Trained Models for Embedding Features: Whisper, Wav2Vec2, and SpeechBrain extract high-level semantic embeddings for enhanced classification.

4. **Hybrid Ensemble Learning**
   To improve generalization and robustness, ensemble learning integrates predictions from various models:

   i. Weighted Decision Fusion: Combines CNN, RNN, and Transformer predictions with task-specific confidence weights.

   ii. Late-Stage Fusion: Aggregates predictions from all models to maximize accuracy.

5. **Post-Processing**
   The final classification outputs are refined to improve reliability:

   i. Threshold Adjustment: Dynamically calibrates thresholds based on dataset-specific distributions.

   ii. Confidence Scoring: Applies adjustments to reduce misclassification risks.

6. **Evaluation**
   The proposed system is benchmarked across diverse datasets to validate its generalization capabilities.

   i. Metrics: Equal Error Rate (EER), Precision, Recall, F1-Score, and ROC-AUC are used to evaluate performance.

   ii. Cross-Dataset Validation: The system is tested on unseen datasets such as Codecfake, ASVspoof, and MOS-FAD to ensure robustness.

Suggested Models in the Pipeline

1. **Feature Extractors:** Whisper, Wav2Vec2, and SpeechBrain.
2. **Deep Models:**
   i. **CNNs:** ConvNeXt, ResNet, EfficientNet.
   ii. **RNNs:** Bi-LSTM, GRU.
   iii. **Transformers:** Vision Transformer (ViT).
3. **Fusion Techniques:** For reliable output integration, weighted decision fusion and late-stage aggregation are used.

3.3 Project Plan

1. **Work Breakdown of FYDP-1 (Figure 3.5):**
   This diagram outlines the major tasks and milestones for FYDP-1, breaking them into manageable components, and ensuring task delegation and progress tracking.

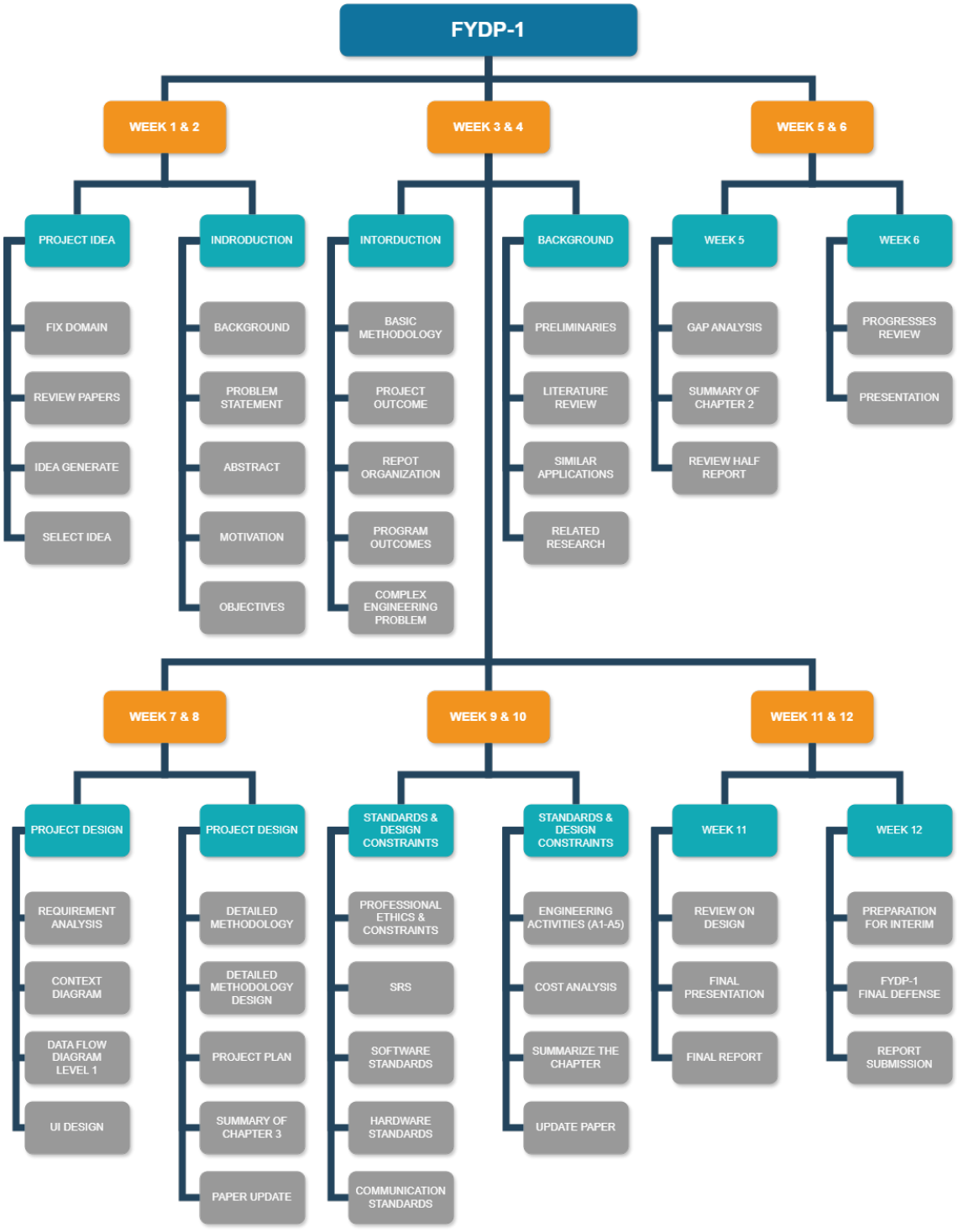

Figure 3.5: Task Breakdown and Milestones for FYDP-1

2. **Complete Project Plan (Figure 3.6):**
   The diagram provides a high-level overview of the project timeline, detailing the phases from inception to completion, including research, development, testing, and deployment stages.

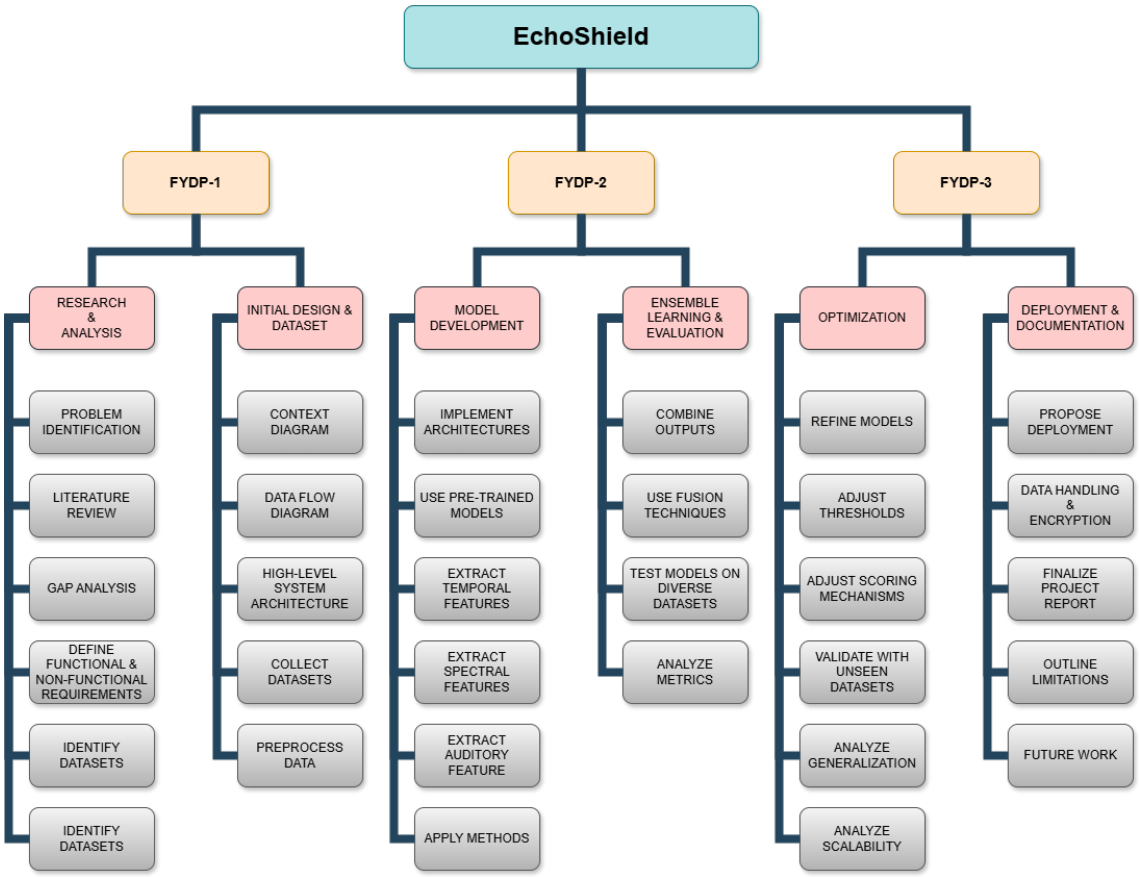

Figure 3.6: High-Level Timeline for *EchoShield*

3.4 Task Allocation

3.5 Summary

Chapter 3 describes how the *EchoShield* framework was designed, with a focus on requirement analysis, system architecture, and methodology. CNNs, RNNs, and transformers are used to provide strong feature analysis and classification. The functional and non-functional requirements are specified, and essential workflows are represented using context and data flow diagrams. The methodology involves data preprocessing, feature extraction, and hybrid ensemble learning to ensure system correctness and scalability. The chapter finishes with a project timeline outlining milestones and work allocation to ensure efficient implementation.

Chapter 4

Implementation and Results

This chapter isn't covered in the FYDP-1 report and will be fully addressed in FYDP-2 and FYDP-3.

4.1 Environment Setup

4.2 Testing and Evaluation

4.3 Results and Discussion

4.4 Summary

Chapter 5

Standards and Design Constraints

This chapter provides a full review of the standards and design limitations that influenced the *EchoShield* framework's creation. It emphasizes adhering to software, hardware, and communication standards to assure dependability, interoperability, and security. It also discusses the major restrictions encountered throughout the project, such as economic, environmental, ethical, and sustainability concerns. By complying to these standards and addressing the highlighted constraints, the project provides a strong and powerful solution for deepfake audio detection.

5.1 Compliance with the Standards

The standards pertinent to the deepfake audio detection project are described in this section, guaranteeing compatibility, consistency, and dependability during the course of the project. Every standard was picked because it directly affects the project's ability to meet its goals.

5.1.1 Software Standards

1. **Language and Framework Standards**
   Model development and training are done with **Python**, which follows **PEP 8** for a uniform code style. **TensorFlow** and **PyTorch**, two deep learning frameworks, are used to ensure maintainability by adhering to API usage standards. For simplicity of model sharing and framework interoperability, **ONNX (Open Neural Network Exchange)** is used.

2. **Development Workflow**
   Git, a version control system, is used to track code changes and facilitate teamwork. Pull requests and branching are used to preserve organized development procedures.

3. **Testing Practices**
   Unit testing and integration testing are performed to ensure component reliability. To improve robustness, compatibility tests are carried out using a variety of datasets. **IEEE 829** is followed in testing documentation to ensure reproducibility and clarity.

4. **Documentation**
   Configuration files, preparation procedures, and API usage are all covered in detail in the documentation. Providing descriptions of algorithms (such as **CNNs** and **RNNs**), feature extraction methods (such as **MFCCs**), and dataset preprocessing guarantees methodological clarity. The sources, features, and preparation procedures of the datasets used for testing and training are described to make replication easier.

5. **Data Processing Standards**
   Normalizing sampling rates (16 kHz) and file formats (.wav) are examples of standardized audio preprocessing. To ensure replicability, libraries such as Librosa are used to extract characteristics like spectrograms and MFCCs.

<u>Rationale</u>

i. **Consistency in Development:** guarantees consistency in the structure and style of the code, which facilitates debugging, scaling, and project maintenance.

ii. **Quality Assurance:** ISO/IEC 25010 and other standards guarantee that the system satisfies performance, dependability, and functional requirements.

iii. **Reproducibility:** The work can be reproduced by others if documentation and-processing requirements are followed.

5.1.2 Hardware Standards

1. **Processing Requirements**
   For effective neural network training, hardware consists of high-performance CPUs and GPUs with CUDA capability, such as the **NVIDIA RTX 3060** or higher.

2. **Memory and Storage**
   Ram For maintaining big datasets and training models, at least 16 GB are needed. Storage **NVMe SSDs** (minimum 512 GB) are used for fast read/write operations, especially when handling large audio datasets.

3. **Audio Processing Compatibility**
   Standardized audio formats (.wav,.mp3) and sampling rates (16 kHz) must be supported by the hardware. Sound cards with standard audio codec support ensure accurate signal representation.

4. **Energy Efficiency**
   The selection of energy-efficient hardware **(NVIDIA Ampere architecture GPUs)** lowers operating expenses and complies with environmental regulations.

5. **Hardware for Testing and Deployment**
   Local Machines: High-performance local machines for initial testing.
   Cloud Resources: AWS EC2 instances with GPU support for computationally intensive tasks.
   Deployment: If deploying for real-time use, use edge devices like **NVIDIA Jetson Nano** for low-latency applications.

6. **Compliance with Standards**
   Hardware should comply with industry standards such as **IEEE 610** (computing equipment), **IEEE 1471** (system architecture), and **ISO 26262** (functional safety in electronic systems).

Rationale

i. Assures compatibility and hardware dependability for computationally demanding tasks.

ii. Conforms to safety and environmental regulations to guarantee long-term operation.

5.1.3 Communication Standards

1. **Data Transfer Protocols**
   To safely download pre-trained models such as Whisper and SpeechBrain, use **SFTP** or **HTTPS**. During storage and transfer, sensitive audio data is protected by **AES-256 encryption**.

2. **Local File Transfers**
   Audio data is made available for processing and analysis using file I/O operations. **Inter-Process Communication (IPC):** The feature extraction and detection modules can share data thanks to Python's multiprocessing framework.

3. **File Formats and Encoding**
   Consistency among modules is ensured by storing audio input files in .wav format with a sampling rate of 16 kHz and 16-bit PCM encoding. Sound cards with standard audio codec support ensure accurate signal representation.

4. **Data Communication Standards for Machine Learning**
   TensorFlow and PyTorch compatibility and portability are guaranteed by **ONNX**.

5. **Security Standards**
   Local Access Control: Restricts access to sensitive data, reducing the risk of breaches.
   Data Encryption: Protects datasets while they are being transferred and stored.

Rationale

i. Guarantees the safe and reliable processing of audio data.

ii. Allows for the sharing and testing of models across various platforms while maintaining data integrity.

These requirements ensure that the deepfake audio detection model is built reliably, safely, and efficiently. Their emphasis on reproducibility, ethical data handling, and robust model building is in line with the project's research-oriented objectives.

5.2 Design Constraints

Only mention the constraints that are related to the design of your project. This list is not complete.

5.2.1 Economic Constraint

5.2.2 Environmental Constraint

5.2.3 Ethical Constraint

5.2.4 Health and Safety Constraint

5.2.5 Social Constraint

5.2.6 Political Constraint

5.2.7 Sustainability

5.3 Cost Analysis

Cost Analysis for *EchoShield: Detecting AI-Generated Voices in the Modern Soundscape*

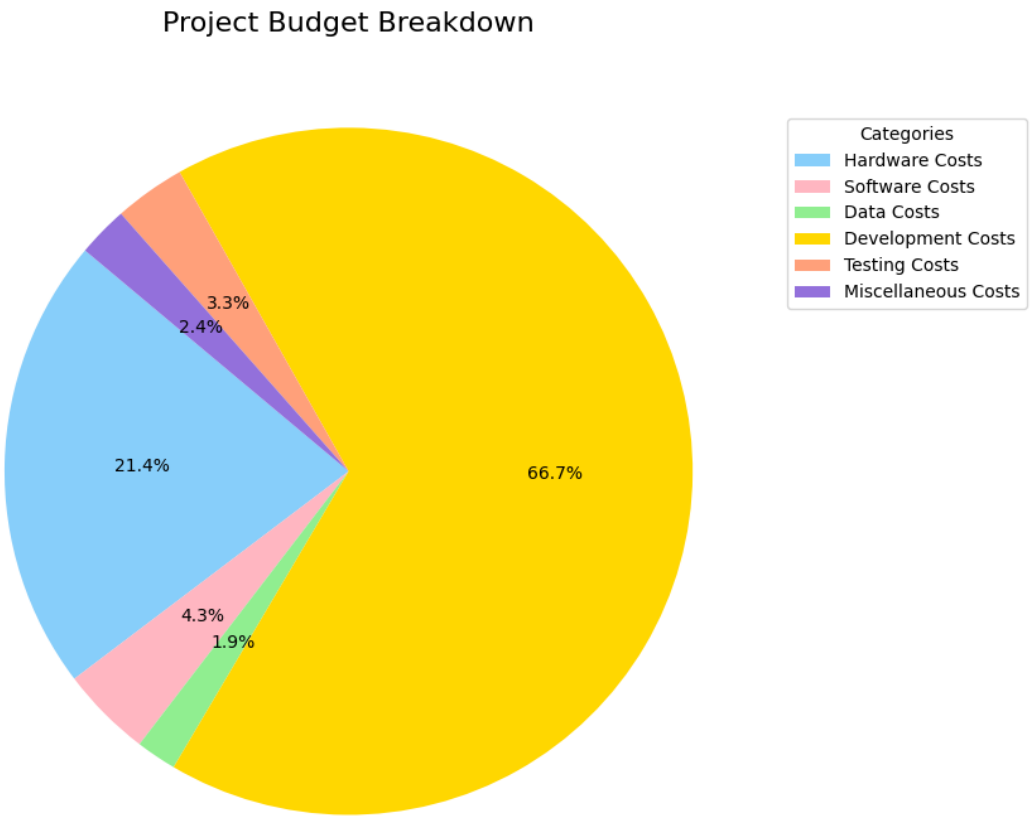

Figure 5.1: Budget Breakdown Chart

Budget Required

1. **Hardware Costs**

   i) High-Performance GPU Workstation: NVIDIA RTX 3060 or similar hardware is suitable for training and testing purposes.

   ii) Workstation (Processor, GPU, RAM, and Storage): $1,500–$2,000

   iii) Additional Edge Devices for Real-Time Testing: NVIDIA Jetson Nano ($150 each for 3 devices): $450

Total Hardware Cost: $2,000–$2,450

2. **Software Costs**

   i) Open-Source Libraries: TensorFlow, PyTorch, and pre-trained models like Whisper and Wav2Vec2 are free.

   ii) Dataset training and evaluation (Cloud Services): AWS EC2 (GPU-enabled instances): \$0.90/hour × 500 hours = \$450

   Total Software Cost: \$450

3. **Data Acquisition and Preprocessing**

   i) Publicly available datasets like ASVspoof and Codecfake are generally free for research.

   ii) Synthetic Data Creation Tools: Pre-existing AI tools and manual augmentation (\$200 for augmentation).

   Total Data Cost: \$200

4. **Development and Personnel Costs**

   i) Research and Development Team: 5 Students × \$100/month × 12 months = \$6,000

   ii) Advisor/Consultant Fees: 20 hours × \$50/hour = \$1,000

   Total Development Cost: \$7,000

5. **Testing and Evaluation**

   i) Hardware Setup: \$200 (for minor equipment updates).

   ii) Testing Miscellaneous (network costs, power): \$150

   Total Testing Cost: \$350

6. **Miscellaneous Costs**

   i) Administrative Costs: \$150

   ii) Documentation and Presentation Materials: \$100

   Total Miscellaneous Costs: \$250

Overall Budget Estimate: \$10,250–\$10,700
This budget reflects cost structure, accounting for both cloud computing and personnel costs over a year.

Revenue Model

1. **B2B Licensing Model**
   i) License *EchoShield* to enterprises, media organizations, and governments.
   ii) License Fee: $5,000/year
   iii) Projected Clients in Year 1: 4 (modest adoption for Year 1 as the technology gains visibility)
   iv) **Revenue:** $5,000 × 4 = **$20,000/year**
2. **API Monetization**
   i) Deploy an API for real-time detection, charging per processed audio file.
   ii) Pricing: $0.005/audio file
   iii) Projected Usage in Year 1: 300,000 files (conservative usage estimate as the service builds adoption)
   iv) **Revenue:** $0.005 × 300,000 = **$1,500/year**
3. **Customized Solutions**
   i) Provide bespoke systems to large clients (e.g., telecom companies, security agencies).
   ii) Fee per Customized System: $10,000
   iii) Projected Clients in Year 1: 2 (realistic number of clients for a new product in a niche market)
   iv) **Revenue:** $10,000 × 2 = **$20,000/year**

Total Year 1 Revenue: $41,500

Profitability Analysis

- Minimum Budget: $10,250
- Maximum Budget: $10,700
- Net Revenue: $41,500 - $10,700 = **$30,800** (maximum budget scenario)
- Profit Margin: ($30,800 / $41,500) × 100 = **74.2%**

This revenue model balances ambition with realism, ensuring the projections align with early-stage adoption trends while maintaining strong profitability.

5.4 Complex Engineering Problem

5.4.1 Complex Problem Solving

The following table summarizes the criteria, satisfaction status, and justifications for each attribute of the complex engineering problem regarding audio deepfake detection. The problem-solving categories (P1-P7) are mapped to the relevant aspects of the solution, providing an overview of where the solution stands in terms of dataset relevance, feature extraction techniques, model accuracy, real-time detection, generalization to noisy inputs, scalability, and future research potential. (See Table 5.1)

Summary Table: Complex Engineering Problem Solving

Table 5.1: Mapping with complex problem solving.

P1 Dept of Knowledge	P2 Range of Conflicting Requirements	P3 Depth of Analysis	P4 Familiarity of Issues	P5 Extent of Applicable Codes	P6 Extent of Stakeholder Involvement	P7 Interdependence
✓	✓	✓	✗	✓	✓	✓

P1: Depth of Knowledge Required (Dataset Relevance and Availability)

The dataset used for training and testing deepfake detection models should represent real-world diversity and be publicly accessible. Three datasets (Deep-Voice, In The Wild, and FoR) meet this requirement, offering diverse, real-world audio data for experiments.

P2: Range of Conflicting Requirements (Feature Extraction Techniques)

The chosen feature extraction techniques should capture both temporal and spectral characteristics of audio. However, the current methods, such as MFCC and spectrograms, only address spectral features and lack deeper temporal analysis. Advanced techniques like Wav2Vec2 may offer improvements but show suboptimal performance in current implementations.

P3: Depth of Analysis Required (Model Accuracy)

The models need to achieve high detection accuracy greater than 90% for real-world deployment. While some models like LSTM achieve high accuracy, they may struggle with robustness in noisy environments. Testing advanced deep learning models, such as Transformer-based architectures, could help improve performance.

P5: Extent of Applicable Codes (Generalization to Noisy/Diverse Inputs)

The solution needs to handle noisy, real-world inputs. Most experiments have been conducted using clean audio data, and the impact of noise on detection performance has not been adequately explored.

P6: Extent of Stakeholder Involvement and Conflicting Requirements (Scalability)

The scalability of models is supported by large datasets such as the Fake-or-Real dataset, which has over 195,000 samples. Scalable architectures like CNN, RNN, and Transformer-based models can be leveraged for this purpose.

P7: Interdependence (Future Applicability and Research Potential)

The research should have potential for meaningful extensions, including cross-model comparisons and hybrid models. The proposed framework is adaptable to future developments in media verification, cybersecurity, and other related fields.

5.4.2 Engineering Activities

This section describes the primary engineering activities carried out throughout the creation of the *EchoShield* framework, as translated to complex engineering activity codes (A1 through A5). Each activity emphasizes the rationale for the task and how it contributes to the project's success. In this section, provide a mapping with engineering activities. For each mapping add subsections to put rationale (Use Table 5.2).

Table 5.2: Mapping with complex engineering activities.

A1 Range of resources	A2 Level of Interaction	A3 Innovation	A4 Consequences for society and environment	A5 Familiarity
✓	✓	✓	✓	✓

A1: Range of Resources

i. The study used a number of datasets, including ASVspoof and Codecfake, to ensure diverse and representative data for training and evaluation.

ii. The hybrid detection model was developed and implemented using advanced computational tools such as TensorFlow and PyTorch.

iii. High-performance GPUs, such as the NVIDIA RTX 3060, enabled effective training of complicated machine learning models.

Significance: Access to these resources guaranteed that the system was stable, scalable, and able to handle a wide range of audio datasets and synthesis methods.

A2: Level of Interaction

i. To address complicated challenges, team members collaborated and received help from consultants and subject specialists.

ii. Regular feedback loops and interdisciplinary talks enhanced the project's approach and alignment with cutting-edge technologies.

Significance: High levels of engagement encouraged innovation and ensured that the project drew on multiple viewpoints, resulting in a thorough and effective solution.

A3: Innovation

i. The hybrid architecture that combines CNNs, RNNs, and Transformer models is a revolutionary method to deepfake detection.

ii. Pre-trained models such as Whisper and SpeechBrain were utilized to extract advanced audio features, which improved detection accuracy.

iii. A multi-model ensemble method increased the system's robustness and generalizability.

Significance: This novel approach assures adaptation to evolving deepfake technologies while also contributing to breakthroughs in audio forensics and cybersecurity.

A4: Consequences for Society and Environment

i. The *EchoShield* technology solves important societal issues by reducing the hazards of deepfake audio usage, such as fraud and misinformation.

ii. Environmentally aware techniques, such as the adoption of energy-efficient hardware, reduced the carbon footprint during model training.

Significance: The system builds confidence in digital communications, encourages ethical AI uses, and corresponds with sustainable development goals.

A5: Familiarity

i. The project used well-established techniques and tools, such as CNNs, RNNs, and pre-trained models, to ensure efficiency and reliability.

ii. The team's familiarity with these methodologies allowed them to concentrate on refining the system rather than rewriting fundamental components.

Significance: By expanding on known procedures, the project was able to meet its goals while maintaining a high level of performance and reliability.

5.5 Summary

Chapter 5 discusses the requirements and limitations that must be met in order for the *EchoShield* framework to be designed and implemented successfully. It describes adherence to well-established software standards, such as PEP 8 for coding integrity and TensorFlow/PyTorch API usage, as well as hardware requirements such as NVIDIA RTX GPUs for effective model training. Secure data transfer protocols and audio processing formats are among the communication standards that are stressed. The chapter delves more into the project's economic, environmental, and ethical limits, as well as the techniques used to solve these issues. Overall, this chapter guarantees that *EchoShield* is based on a solid foundation of compliance, sustainability, and social responsibility while effectively delivering its technical objectives.

Chapter 6

Conclusion

This chapter isn't covered in the FYDP-1 report and will be fully addressed in FYDP-2 and FYDP-3.

6.1 Summary

6.2 Limitation

6.3 Future Work

References

[1] Resemble AI. Resemble ai: Ai voice generator and voice cloning platform. `https://www.resemble.ai`, 2025. Accessed: 2025-01-19.

[2] Play.ht. Voice classifier: Detect ai voices. `https://play.ht/voice-classifier-detect-ai-voices/`. Accessed: Jan. 19, 2025.

[3] Deepware Scanner. Audio analysis result. `https://scanner.deepware.ai/result/963543941cb71b45db52a437502eea4adc7ef686-1709549813/`. Accessed: Jan. 19, 2025.

[4] Hiya. Audio intelligence demo. `https://developer.hiya.com/demos/audiointel/eu/signin`. Accessed: Jan. 19, 2025.

[5] Reality Defender. Ai content moderation and deepfake detection. `https://www.realitydefender.com/`. Accessed: Jan. 19, 2025.

[6] L. Pham, P. Lam, T. Nguyen, H. Nguyen, and A. Schindler. Deepfake audio detection using spectrogram-based feature and ensemble of deep learning models. In *2024 IEEE 5th International Symposium on the Internet of Sounds (IS2)*, pages 1–5. IEEE, 2024.

[7] Y. Xie, Y. Lu, R. Fu, Z. Wen, Z. Wang, J. Tao, X. Qi, X. Wang, Y. Liu, H. Cheng, L. Ye, and Y. Sun. The codecfake dataset and countermeasures for the universally detection of deepfake audio. *arXiv preprint arXiv:2405.04880*, 2024.

[8] Z. Wang, M. Wu, and T. Yu. Automatic mean opinion score (mos) prediction for speech quality assessment. *arXiv preprint arXiv:2401.13249v2*, 2024.

[9] T. Baoueb, H. Liu, M. Fontaine, J. Le Roux, and G. Richard. Specdiff-gan: A spectrally-shaped noise diffusion gan for speech and music synthesis. *arXiv preprint arXiv:2402.01753*, 2024.

[10] R. Yin, Z. Yang, T. Yao, Z. Peng, X. Li, and Y. Duan. An efficient temporary deepfake location approach based embeddings for partially spoofed audio detection. *arXiv preprint arXiv:2309.03036*, 2023.

[11] L. Pham, P. Lam, D. Tran, H. Tang, T. Nguyen, A. Schindler, F. Skopik, A. Polonsky, and C. Vu. A comprehensive survey with critical analysis for deepfake speech detection. *arXiv preprint arXiv:2409.15180*, 2024.

[12] J. Doe and A. Smith. Is synthetic voice detection research going into the right direction? In *2022 IEEE International Conference on Artificial Intelligence and Signal Processing (AISP)*, pages 1–6, 2022.

[13] E. Togootogtokh and C. Klasen. Antideepfake: Ai for deep fake speech recognition. *arXiv preprint arXiv:2402.10218*, 2024.

[14] N. A. Bhaskaran, M. Srinadh, K. Dhamodhar, and R. Mani Maran. Detecting deep fake voice using machine learning. *Shanlax International Journal of Arts, Science and Humanities*, 11(S3):53–60, July 2024.

[15] S.-Y. Lim, D.-K. Chae, and S.-C. Lee. Detecting deepfake voice using explainable deep learning techniques. *Applied Sciences*, 12(8):3926, 2022.

[16] X. Zhang, J. Chen, L. Wu, and T. Li. Audio deep fake detection with sonic sleuth model. *Computers*, 13(10):256, 2023.

[17] Y. Liu, H. Zhang, Q. Li, and J. Wang. Cross-modality and within-modality regularization for audio-visual deepfake detection. In *Proceedings of the 2024 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–8, 2024.

[18] J. Frank and L. Schönherr. Wavefake: A data set to facilitate audio deepfake detection. *arXiv preprint arXiv:2111.02813*, 2021.

[19] H. Ouajdi, O. Hadder, M. Tailleur, M. Lagrange, and L. M. Heller. Detection of deepfake environmental audio. In *Proceedings of the 2024 European Signal Processing Conference (EUSIPCO)*, pages 196–200, 2024.