ADSIFT - AN INTELLIGENT AUDIO CLASSIFIER FOR MUSIC AND ADVERTISEMENT CLASSIFICATION IN FM RADIO USING RESNET-18

ABSTRACT

In today's world, a system that distinguishes between music and advertisements is essential to improve the FM radio listening experience and minimize interruptions, especially for drivers. Advertisements, though necessary for station revenue, disrupt the listening flow and increase cognitive load and distraction. To address this, an intelligent FM audio classification system called Adsift is proposed.Adsift integrates FFmpeg for live FM streaming, Librosa for audio preprocessing, PyTorch for model training, and employs a ResNet-18 convolutional neural network for feature extraction from mel-spectrograms. The system classifies FM content in real-time, automatically switching to alternate music stations to avoid advertisements. Initial results show high classification accuracy and smooth, uninterrupted music playback, enhancing both user satisfaction and driving safety. Future enhancements include implementing a rolling five-second audio buffer to eliminate the switching delay after ad detection, supporting multilingual advertisement detection, reducing detection latency further, and incorporating reinforcement learning to personalize station selection. Applications of Adsift extend to smart car infotainment systems, personalized radio services, and safer, cognitive-load-aware driving experiences.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

FM	-	Frequency Modulation
CNN	-	Convolutional Neural Network
ResNet	-	Residual Network
API	-	Application Programming Interface
FFmpeg	-	Fast Forward MPEG
UI	-	User Interface
ASGI	-	Asynchronous Server Gateway Interface
HTML	-	HyperText Markup Language
CSS	-	Cascading Style Sheets
NVM	-	Node Version Manager
CUDA	-	Compute Unified Device Architecture
NPM	-	Node Package Manager
JSON	-	JavaScript Object Notation
AUC	-	Area Under the Curve
ROC	-	Receiver Operating Characteristic
SVM	-	Support Vector Machine
LSTM	-	Long Short-Term Memory (a type of Recurrent Neural Network, RNN)
VGG	-	Visual Geometry Group
MFCC	-	Mel-Frequency Cepstral Coefficients

CHAPTER 1
INTRODUCTION

1.1 Background and Project Domain

FM radio continues to be a widely used medium for music and information, particularly among drivers during travel. However, frequent advertisements, necessary for station revenue, often disrupt the flow of music and contribute to increased driver distraction and cognitive load. Traditional FM receivers require manual station switching to avoid commercials, posing safety risks and reducing listening satisfaction. Research highlights that continuous, familiar music input enhances concentration, reduces stress, and improves cognitive performance while driving. Despite advancements in deep learning and audio signal processing, conventional FM systems lack automated content filtering solutions.To address this gap, this project proposes Adsift, an intelligent FM audio classification system. Using real-time audio processing and deep learning models, Adsift aims to detect advertisements and automatically switch to alternate FM channels as shown in figure below (Fig. 1.1), maintaining uninterrupted music playback and improving road safety.

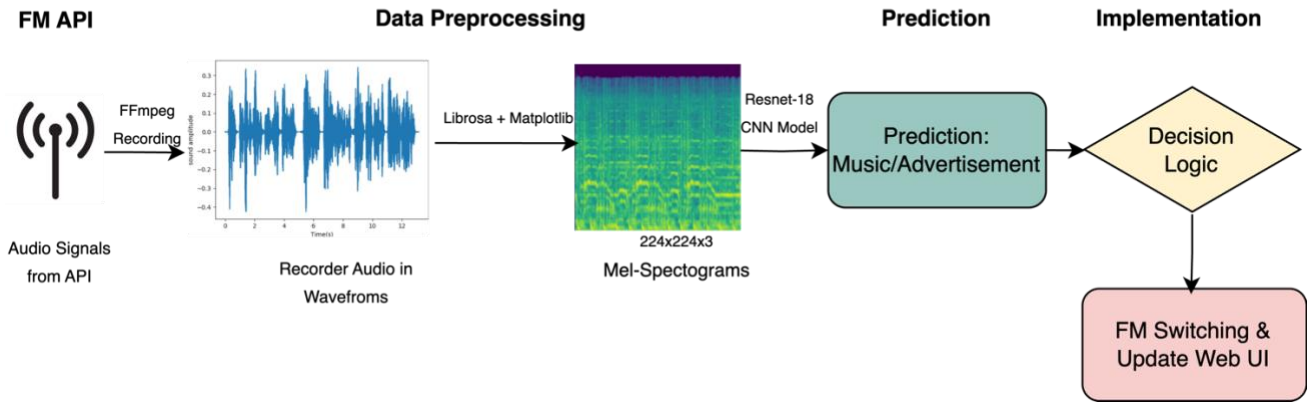

Fig 1.1 Adsift Functional Flow Diagram

1.2 Problem Statement

Current FM radio systems do not provide automated ad-skipping capabilities. Listeners are forced to manually switch stations to avoid advertisements, causing inconvenience and posing safety concerns for drivers. There is a need for an intelligent system that can classify FM content in real-time and ensure seamless, uninterrupted music streaming without manual intervention.

1.3 Objectives

The objectives of this project are:

- Develop a real-time FM audio classifier using CNNs and mel-spectrograms to distinguish music from advertisements.
- Enable dynamic station switching for uninterrupted music based on user preferences.

1.4 Applications:

The Adsift system can be applied in several areas:

- **Smart Car Infotainment Systems:** Enhancing in-vehicle audio systems with automated ad-skipping.
- **Personalized Radio Streaming Services:** Providing dynamic, ad-free music experiences.
- **Public Transportation Entertainment:** Offering seamless music streaming in buses, trains, and taxis.

CHAPTER 2
LITERATURE SURVEY

2.1 Detailed Review of Research Works

Recent years have witnessed significant advancements in audio classification technologies, particularly those aimed at FM radio broadcast analysis, music/speech segmentation, and driver-focused audio experiences. The reviewed papers collectively contribute to understanding the impact of auditory stimuli, optimizing audio classification architectures, and building intelligent systems capable of real-time audio content control.

One of the earliest motivations for intelligent FM radio systems stems from studies examining the impact of music on driving performance. Research such as "Effect of Music on Driving Performance" and "The Influence of Music on Mental Effort and Driving Performance" highlights that music familiarity and tempo significantly affect driver alertness. Fast, unfamiliar, or complex music increases cognitive workload and response times, while familiar, soothing music stabilizes driving behavior. Moreover, "The Effect of Music Familiarity on Driving" further confirms that cognitive load is lower when drivers listen to known songs, stressing the importance of recognizing not just genre, but familiarity in future radio systems.

Given these behavioral influences, the early efforts in radio content management involved manual tuning or simple ML-based station recommendation systems. In "Automatic Tuning of Radio Stations Based on Listener's Preference", machine learning models predict user preferences based on content types (e.g., music vs speech) but faced challenges in real-time classification speed and accuracy. "Understanding the Passive Listeners of FM Radio Stations" surveyed listener behavior, identifying that passive users prefer automated systems that minimize

manual intervention, thereby justifying the demand for intelligent ad-skipping FM systems.

To support such functionalities, various audio content classification models have been proposed. Early solutions like those in "RADIA: Radio Advertisement Detection with Intelligent Analytics" utilized MFCC feature extraction and classical models like SVMs. While achieving high accuracy (~92%), these systems suffered when exposed to noisy backgrounds or overlapping speech and music—a common situation in real-world FM broadcasts.

Recognizing the limitations of manual feature extraction, the field shifted toward deep learning, especially CNN-based architectures. In "CNN Architectures for Large-Scale Audio Classification", a comprehensive comparison demonstrated that deep CNNs (ResNet, VGG, AlexNet) vastly outperform classical models by autonomously learning hierarchical features. Among these, ResNet variants, particularly ResNet-18 and ResNet-50, emerged as frontrunners due to their residual connections that allow deeper network construction without degradation issues.

Specific works like "ResNet-based Sound Classification in Noisy Environments" validated ResNet's robustness under adverse conditions by applying spectral masking and augmentations. Here, ResNet-18 models showed over 5% improvement in classification accuracy over plain CNNs when tested on datasets infused with synthetic noise. This finding reinforces ResNet’s suitability for FM broadcast analysis, where real-time noise is inevitable.

Further studies, such as "Audio-Based Music Classification with a Pretrained CNN" and "Comparison of Pre-Trained CNNs", examined transfer learning. ResNet architectures pre-trained on audio datasets like AudioSet and then fine-tuned on specific FM datasets led to better performance while reducing training costs. Fine-tuning pre-trained networks significantly improves convergence time and accuracy, making it ideal for scenarios like dynamic FM station classification.

Low-latency systems like "SwishNet: A Fast CNN for Audio Classification" were introduced for resource-constrained environments. Although SwishNet offers fast inference and smaller memory footprints, studies report slight trade-offs in noisy environments compared to deeper ResNet models. This observation suggests that for mission-critical systems like ad-skipping in moving vehicles, deeper models remain preferable.

Another notable trend is the creation of synthetic datasets to enhance generalization. In "Artificially Synthesizing Data for Audio Classification and Segmentation", augmentation techniques like pitch shifting, speed changes, and noise addition were applied to enlarge datasets, ultimately improving classifier resilience to unexpected audio patterns common in FM broadcasts.

In the domain of FM advertisement detection, several innovations have emerged. For example, "Detecting Advertising in Radio Using Machine Learning" focused on combining time-domain and frequency-domain feature analyses to segment ads, using hybrid CNN-RF models. "Classification of Voice Content in Radio" extended this by using CNN+LSTM hybrids, where CNNs extract spatial features and LSTMs model temporal dependencies, achieving superior accuracy over plain CNNs.

Complementary to the technical advancements, semantic studies like "Extending Radio Broadcasting Semantics Through Machine Learning" propose enriching FM broadcasts beyond simple music-speech-ad segmentation, towards recognizing moods, genres, or even emotional tones based on audio content. Such semantic enrichment could drive the next generation of intelligent FM radio services.

Finally, domain-specific dataset creation efforts, like "KritiSamhita: A Machine Learning Dataset of South Indian Classical Music Audio Clips", offer specialized resources for fine-grained classification tasks. Although focused on

South Indian classical music, these datasets exemplify how domain-specific data can enhance model precision for real-world applications.

Despite remarkable advancements, gaps remain. Short advertisement snippets, overlapping speech and background music, and dynamic noise conditions continue to challenge classifiers. Further, deployment in low-power environments (such as automotive on-board units) requires balancing between model depth, inference latency, and memory consumption.

In conclusion, the surveyed literature reveals a strong evolution from feature-engineered models to deep CNN-based, transfer learning-enabled, noise-resilient systems, with ResNet-18 and ResNet-50 architectures emerging as the most balanced and effective choices for real-time FM radio content classification and advertisement detection.

2.2 Summary of Literature Survey:

S.No	Research Area	Techniques/Findings	Challenges Addressed
1	Effect of Music on Driving	Familiar music stabilizes driving; complex music distracts	Need for adaptive content control
2	Listener Behavior Analysis	Passive listeners prefer automated ad-skipping	Supports automation need
3	Early ML Models (MFCC+SVM)	MFCC features + classical ML	Poor noise and overlap handling
4	RADIA System	ML-based advertisement detection	Ineffective under overlapping sounds

5	CNN Architectures	CNNs (VGG, ResNet) outperform classical methods	Improved feature learning
6	ResNet Models	ResNet-18/50 enable deeper learning without overfitting	Robust to noise and complexity
7	ResNet in Noisy Environments	Spectral masking + augmentation	Noise-resilient models
8	Transfer Learning	Pre-trained CNNs fine-tuned for FM audio	Faster convergence and higher accuracy
9	SwishNet	Lightweight CNN model	Lower robustness under noise
10	Synthetic Dataset Augmentation	Pitch/Speed/Noise augmentation	Improved model generalization
11	Hybrid CNN+LSTM Models	Temporal modeling improves ad/music segmentation	Better segmentation accuracy
12	RF-CNN Models for Ads	Hybrid random forest + CNN	Enhanced ad block detection
13	Semantic FM Enrichment	Genre/mood/emotion classification	Future expansion beyond ads
14	Domain-Specific Datasets	South Indian classical music dataset (KritiSamhita)	Specialized fine-grained classification

2.3 Inferences Identified from Literature Survey

- Deep learning, especially ResNet architectures, now dominate audio classification research for FM radio.
- Traditional machine learning models are no longer sufficient for handling noisy or overlapping FM broadcasts.
- Listener behavior studies show a strong preference for automated systems that skip ads and continue preferred content without user input.
- Transfer learning (fine-tuning pre-trained CNNs) is very effective, saving time and boosting performance.
- Data augmentation strategies are necessary to make models robust for real-world noisy conditions.
- Hybrid architectures (CNN+LSTM) are better for recognizing audio that changes over time.
- Lightweight models like SwishNet are helpful for mobile devices but still slightly weaker in noisy situations.
- Short ad snippets and overlapping content are open problems needing more advanced solutions.
- Real-time deployment must balance between model complexity (depth) and system resource limitations (speed/memory).
- Semantic enrichment (mood, genre detection) is a new future direction for smarter FM radio systems beyond basic ad-skipping.

CHAPTER 3
SYSTEM ARCHITECTURE

The Adsift system architecture is designed to ensure seamless, real-time audio classification and advertisement filtering in FM radio streams. It combines machine learning, real-time audio streaming, and a user-friendly web interface. The architecture leverages multiple technologies and frameworks to ensure smooth processing, secure authentication, and efficient interaction between the user and the system.

3.1 Audio Classification Model

The core of the Adsift system is the ResNet-18 variant Convolutional Neural Network (CNN), which classifies incoming FM audio streams into music and advertisements. The model works with mel-spectrograms, a time-frequency representation of the audio signal, which is input into the ResNet-18 model for classification. The figure below (Fig. 3.1) illustrates the architecture of the ResNet-18 model.

- **Input Layer**: The model takes mel-spectrograms as input. These are generated from FM audio streams using Librosa and FFmpeg, which are Python libraries used for audio preprocessing.
- **Residual Blocks**: The core architecture consists of multiple residual blocks, which allow the model to retain important features and solve the vanishing gradient problem.
- **Softmax Output**: The final layer of the model provides a softmax output, classifying the input audio into either music or advertisement.

The model is trained using a cross-entropy loss function and optimized with the Adam optimizer. The classification is real-time, as the model continuously processes incoming audio data and provides classification results.

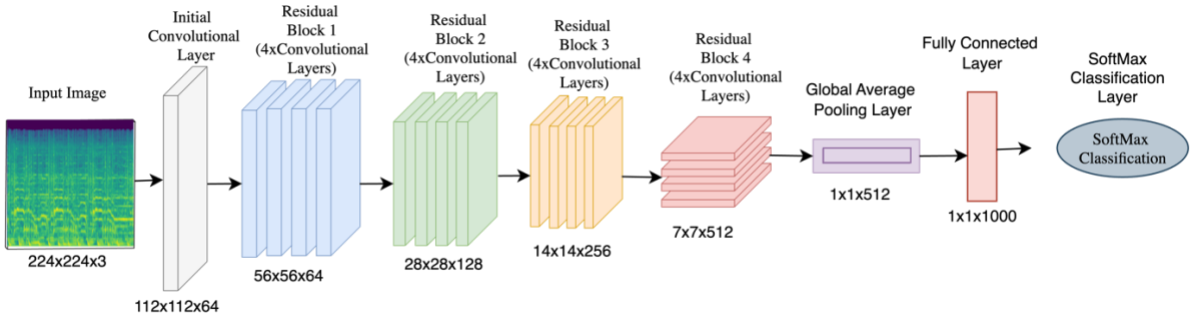

Fig 3.1 Model Architecture (Resnet - 18)

3.2 Audio Preprocessing and Mel-Spectrogram Generation

Before the audio data enters the classification model, it must undergo preprocessing. This is achieved using Librosa and FFmpeg to convert the FM audio streams into mel-spectrograms.

1. **Audio Stream Division**: The FM audio streams are recorded and divided into 5-second frames.
2. **Mel-Spectrogram Conversion**: These frames are converted into mel-spectrograms using Librosa.
3. **Normalization**: The generated mel-spectrograms are normalized with mean and standard deviation values for better model performance:
    - Mean: [0.456, 0.406, 0.225]
    - Standard Deviation: [0.229, 0.224, 0.225]

Once these steps are complete, the spectrograms are fed into the ResNet-18 model for classification as shown in figure (Fig. 3.2).

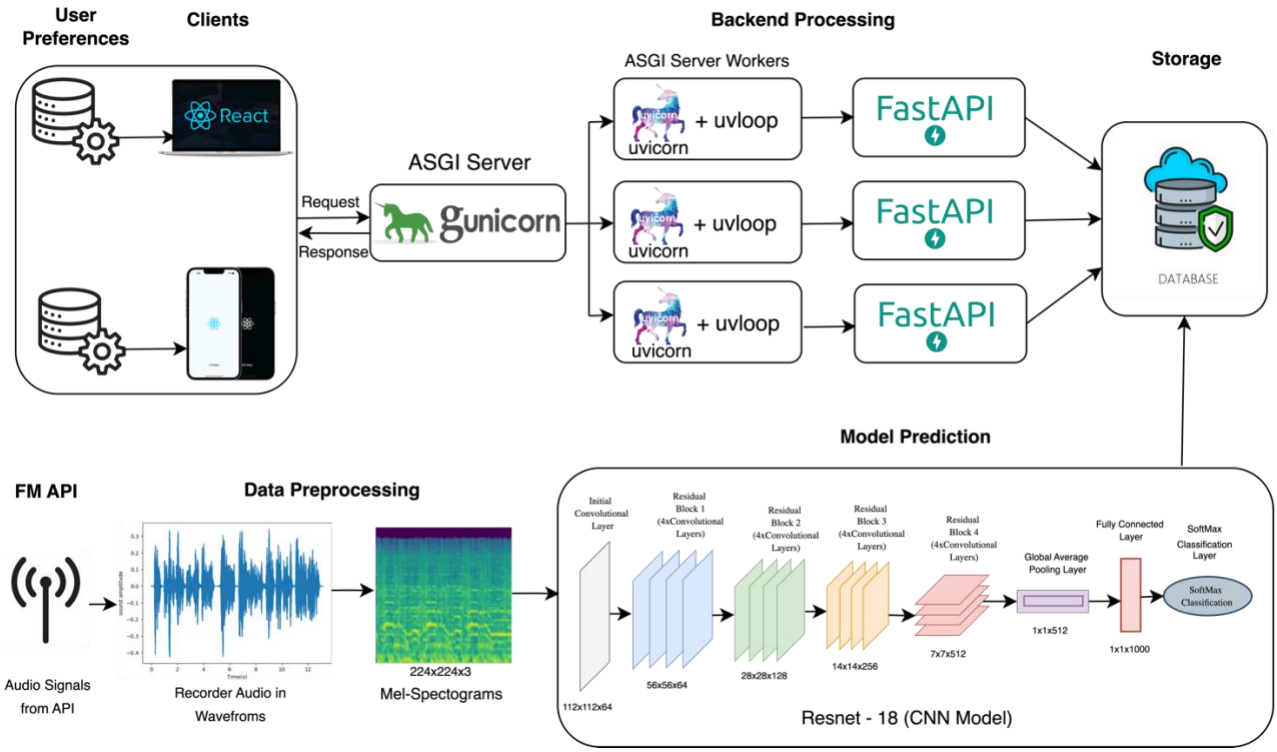

Fig 3.2 System Architecture

3.3 Client-Server Authentication Infrastructure

To secure the system, Adsift implements an authentication infrastructure based on a client-server architecture. The FastAPI framework handles authentication, user registration, and login management as shown in above figure (Fig. 3.2).

1. **Authentication**: When users attempt to access the system, their credentials are verified by the server, which ensures that only authorized users can interact with the system.
2. **User Preferences**: After successful authentication, the system tailors FM station suggestions to the user's preferences.

The authentication system is powered by FastAPI, which is known for its high performance, allowing quick API responses for authentication and access control.

3.4 Real-Time FM Streaming and Classification

The FM Web Interface is the central platform for user interaction. It is built using React.js, allowing users to control FM stations and see real-time classification results. The FM streaming is powered by the Radio Browser API, which provides access to live telecasts.

1. **Real-Time Audio Streaming**: The backend services continuously stream FM radio stations using FFmpeg. The streams are processed and converted into mel-spectrograms for classification.

   The live FM radio audio is fetched using the Radio Browser API and played in the React.js frontend using the HTML audio tag. This allows the user to hear the live FM stream while the system classifies the content.

2. **Classification and Station Switching**: As each frame is classified by the ResNet-18 model, the system automatically switches to a different FM station if an advertisement is detected.
3. **User Interaction**: The React.js frontend provides an intuitive interface for users to interact with the system, view classification results, and switch stations.

3.5 Backend Services and Real-Time Classification with FastAPI

The backend of the Adsift system is implemented using FastAPI, which serves as the API layer for communication between the web interface and the server. The backend is optimized to handle real-time audio processing, classification, and user requests as shown in figure (Fig. 3.2).

1. **Uvicorn with Uvloop and Gunicorn**: The backend is hosted using Uvicorn, a fast ASGI server. Uvloop is used to optimize event loops for better concurrency, ensuring low-latency responses. Gunicorn serves as a multi-process server to handle multiple concurrent user requests effectively.
2. **Audio Processing**: The backend continuously receives FM audio streams, processes them into mel-spectrograms, and sends them to the ResNet-18 model for classification.
3. **Automatic Station Switching**: If an advertisement is detected, the backend triggers automatic switching to a different FM station, ensuring uninterrupted music playback for the user.

3.6 Real-Time Logging and Monitoring

To ensure smooth system operation, real-time logging is implemented for tracking user actions, classification results, and station-switching events.

1. **User Actions**: All user actions, such as logging in and switching stations, are logged for auditing and system diagnostics.
2. **Classification Results**: Each classification output (whether the segment is music or advertisement) is logged for performance tracking.
3. **Station Switches**: Events when the system switches FM stations are logged for monitoring.

The logs are managed and analyzed for continuous improvement of system performance and user experience.

3.7 Deployment with Ngrok and Scalability

For deployment and remote access, Ngrok is used to expose the backend API to the internet. Ngrok creates secure tunnels to the local server, allowing users to interact with the system from anywhere.

1. **Ngrok**: By using Ngrok, the FastAPI backend can be exposed to the internet for public access, enabling users to authenticate and interact with the FM system.
2. **Scalability**: The system is designed to handle multiple concurrent audio streams through parallel processing. The backend services are capable of managing multiple FM stations simultaneously, using multi-threading and batch inference strategies.

The system can also be scaled horizontally by deploying it in cloud environments, making it adaptable to growing user demands.

CHAPTER 4
SYSTEM REQUIREMENTS

4.1 Hardware Requirements

- **Processor (Windows):** Intel Core i3/i5/i7 or AMD Ryzen 5/7 (or equivalent) with a base clock speed of 2.0 GHz or higher
- **Processor (Mac):** Apple M1 chip or newer (M1 Pro, M1 Max, M2, or M3)
- **System Type:** 64-bit Operating System (Windows/Linux) or MacOS 12 (Monterey or above)
- **Hard Disk:** Minimum 30 GB free space
- **Memory (RAM):** 8 GB and above (16 GB recommended for deep learning tasks)
- **Monitor:** Full HD (1920×1080) resolution
- **Graphics Card:** NVIDIA GPU (GTX 1650 or higher) with CUDA support (recommended for training on Windows/Linux) or Apple GPU (M1/newer) for MacOS (optional for Intel Macs, eGPU recommended for better performance)

4.2 Software Requirements

- **Operating System**: Windows 10/11, Ubuntu 20.04+, macOS
- **Programming Language**: Python 3.8 and above
- **Frameworks & Libraries**:
    - **Deep Learning**: Torch, Torchvision, Scikit-learn
    - **Audio Processing**: FFmpeg, Librosa
    - **Backend**: FastAPI, uvicorn, gunicorn, Ngork
    - **Frontend**: React.js, HTML, CSS, JavaScript, Node.js
    - **Visualization**: Matplotlib, Seaborn
- **Development Environment**: Jupyter Notebook, VS Code
- **Package Manager**: pip, npm, nvm

CHAPTER 5
SYSTEM DESIGN AND IMPLEMENTATION

The Adsift system is a real-time, intelligent FM radio management platform that leverages machine learning, deep learning, and web technologies to provide users with uninterrupted music playback by filtering out advertisements. This chapter explains the design, module-wise implementation, and the working algorithm behind the project.

5.1 System Overview

The architecture of Adsift consists of three primary layers:

- **Recording and Preprocessing Layer**: Captures FM streams and prepares them for classification.
- **Classification and Decision-Making Layer**: Classifies audio and determines appropriate actions (continue/station switch).
- **Client Interaction Layer**: Interfaces with the user via a dynamic web application.

Each layer communicates with others through APIs, real-time JSON data updates, and event-driven workflows, ensuring minimal latency and maximum responsiveness.

5.2 Project Modules and Algorithms

5.2.1 FM Stream Recording Module

Functionality:

This module is responsible for continuously capturing audio from online FM station URLs at 5-second intervals.

Implementation:

- FFmpeg is used to connect to the stream and record short 5-second audio clips.
- Recordings are saved as .wav files, with filenames indicating station ID and segment number.

Algorithm:

1. Initialize stream URL and configure output paths.
2. Trigger FFmpeg to capture a 5-second audio segment.
3. Save the recording with a uniquely identifiable filename.

5.2.2 Mel-Spectrogram Conversion Module

Functionality:

Converts recorded audio into mel-spectrogram images to feed into the classification model.

Implementation:

- Librosa is used for signal processing.
- Matplotlib saves mel-spectrograms as images after scaling and formatting.

- Images are resized to match the input dimensions expected by the CNN (224x224).

Algorithm:

1. Load the .wav file using Librosa.
2. Compute the mel-spectrogram and apply a logarithmic (decibel) scale.
3. Save the resulting spectrogram as an image.
4. Resize the image appropriately for model input.

5.2.3 Audio Classification Module

Functionality:

Classifies each spectrogram image into two categories: Music or Advertisement.

Implementation:

- ResNet-18, pre-trained on ImageNet and fine-tuned on the FM dataset, performs the classification.
- Images are normalized using standard ImageNet mean and standard deviation values.

Algorithm:

1. Load the trained ResNet-18 model.
2. Preprocess the spectrogram image (resize, normalize).
3. Input the image to the model for inference.
4. Output the predicted label (Music or Advertisement).

5.2.4 Decision-Making and FM Station Switching Module

Functionality:

Controls the behavior of FM station selection based on the classification results.

Implementation:

- Maintains an fm_status dictionary that tracks active stations.
- If an advertisement is detected, switches the FM station automatically.
- Ensures the user always hears music without manual intervention.

Algorithm:

1. Monitor the classification output for each 5-second segment.
2. If an Advertisement is detected:

    i. Remove the current station from the active pool.
    ii. Randomly select and switch to another station.

3. If Music is detected then keep the current station active.
4. Update fm_status with current selections.

5.2.5 Web Interface Module (React.js Frontend)

Functionality:

Provides an intuitive, real-time dashboard for users to interact with the system.

Implementation:

- Built using React.js with asynchronous data fetching from FastAPI.
- Displays station status, classification result (Music/Advertisement), and handles user authentication (Login/Signup).

Features:

1. Real-time station monitoring.
2. Dynamic UI updates based on API responses.
3. Smooth user authentication flow.

5.2.6 Data Storage Module (JSON File Handling)

Functionality:

Maintains a persistent state of FM station activity for real-time monitoring and system recovery after crashes.

Implementation:

- Every 5 seconds, the system updates the fm_status dictionary.
- The dictionary is serialized and saved into a .json file.
- The frontend fetches this file through an API to render real-time updates.

Algorithm:

1. After every classification and decision-making step, update the fm_status.
2. Save the updated dictionary into a .json file using Python's json module.

CHAPTER 6
RESULTS AND DISCUSSION

The suggested Adsift system provides a strong and effective solution for automatic audio classification into advertisements and music using mel-spectrograms as input features and a fine-tuned ResNet-18 model for binary classification.

6.1 Performance Metrics

As clear from Table 6.1, the model was experimented on different splits of datasets—70-30 standard split, 80-20 standard split, and 5-Fold Cross-Validation. Among them, the 80-20 split was the best with accuracy, precision, recall, and F1 score of 97.45%. This clearly indicates that the system is highly efficient in identifying discriminative features between ads and music when trained on a slightly larger training set with nicely separated validation data.

Table 6.1 Performance Metrics for Different Dataset Splits

Dataset Split	Accuracy	Precision	Recall	F1 Score
70-30 Standard Split	96.59%	96.67%	96.59%	96.54%
80-20 Standard Split	97.45%	97.45%	97.45%	97.43%
5-Fold Cross-Validation	96.75%	97.21%	95.07%	96.03%

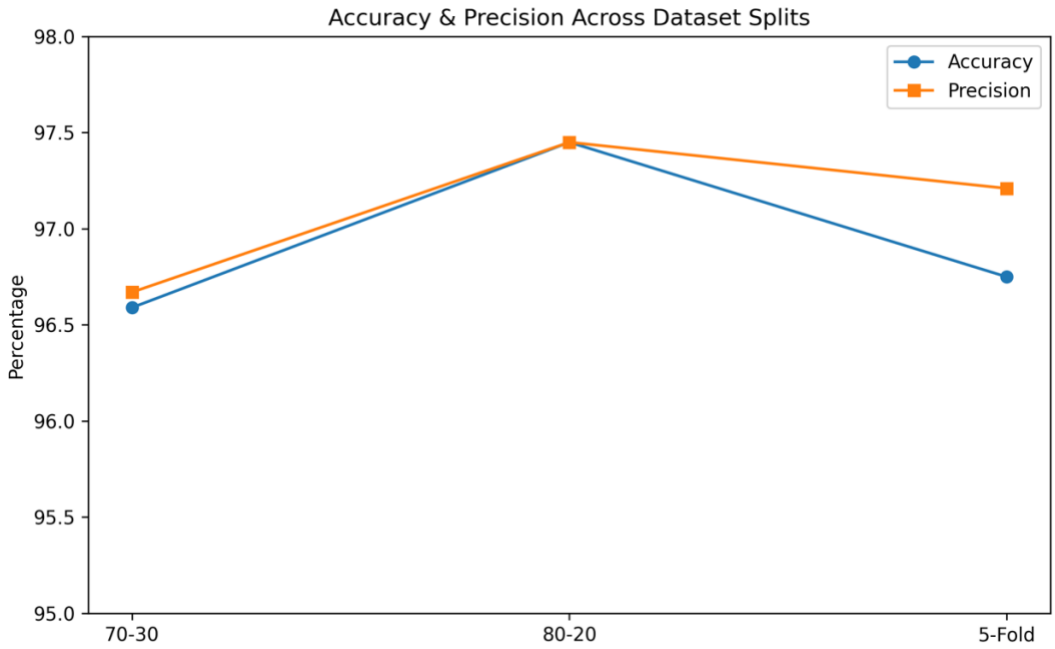

Fig 6.1 Comparing Accuracy and Precision with different Dataset Splits

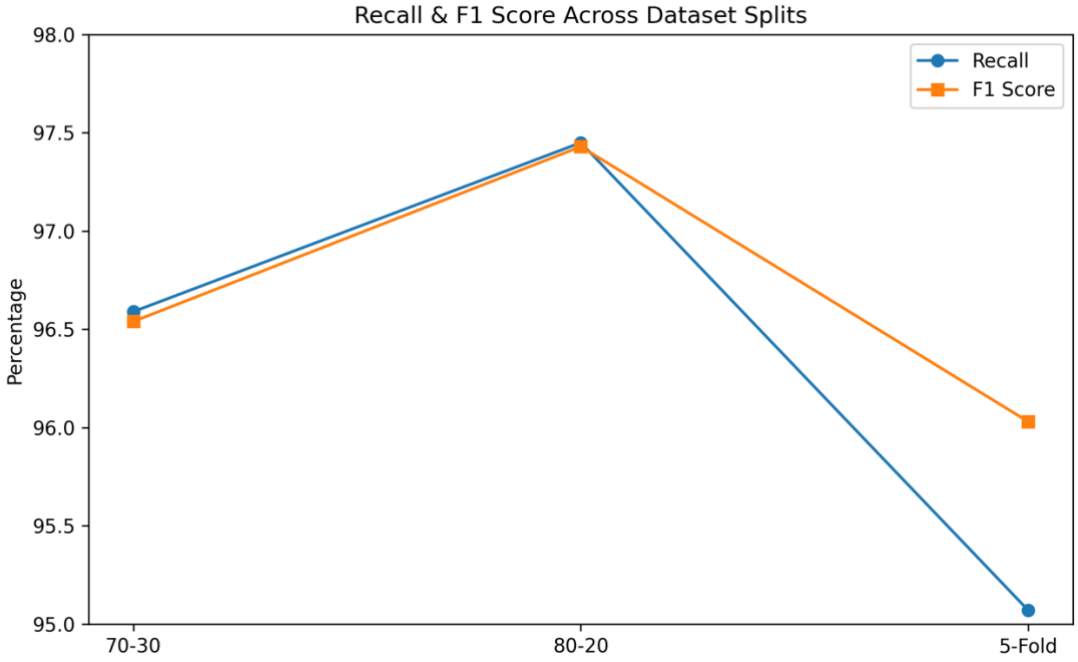

Fig 6.2 Comparing Recall and F1 Score with different Dataset Splits

Fig. 6.1 and 6.2 shows the comparison of accuracy, precision, recall, and F1-score with different dataset splits. Interestingly, the 80-20 split performs better than

others all the time, especially in recall and F1-score. This substantiates the conclusion that a balanced ratio between training and validation data improves both generalization power and consistency in classification in real-world applications.

6.2 Model Training Behaviour

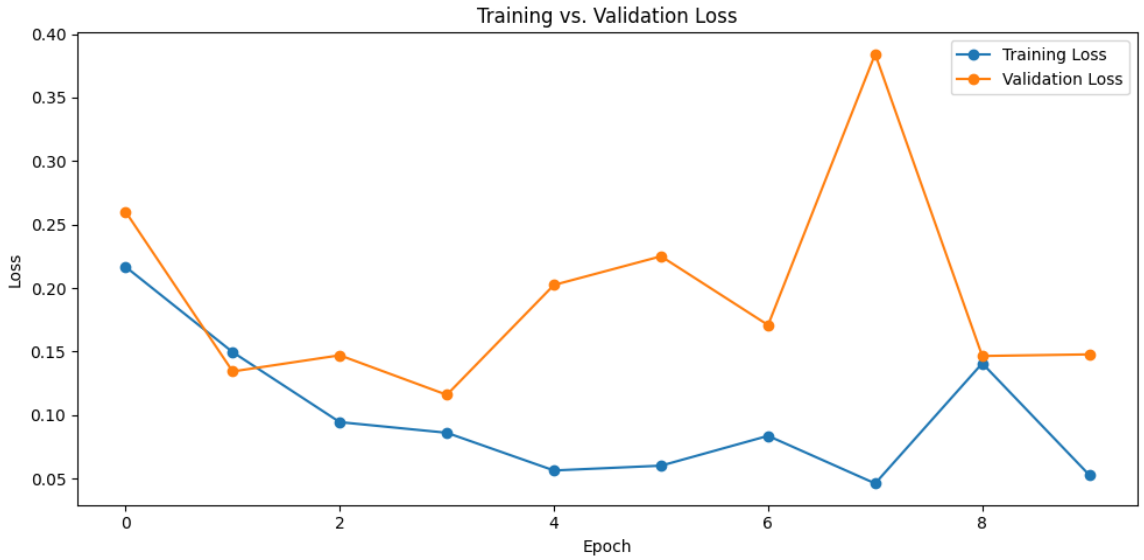

Fig 6.3 Training vs Validation Loss

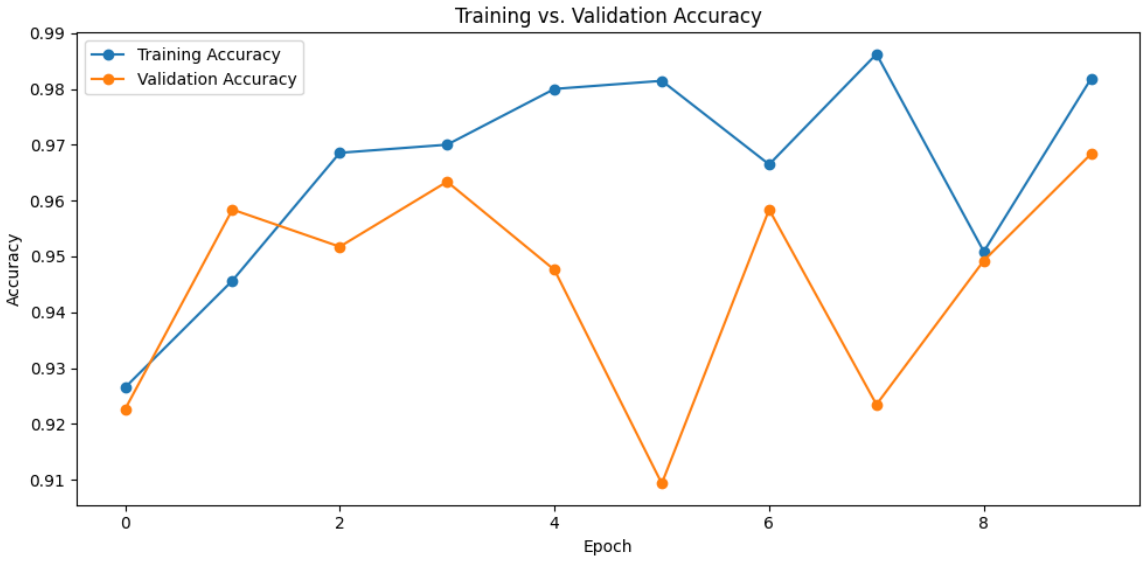

Fig 6.4 Training vs Validation Accuracy

The model training process is depicted through training vs. validation loss and accuracy curves. The convergence of the two curves with minimal overfitting indicates a well-regularized model. The validation loss in Fig. 6.3 experiences slight oscillations but remains largely in proximity to the training loss, implying a good fit without much variance. The training vs. validation accuracy curves in Figure 6.4 also stay very close, indicating stable learning and good predictive capability on unseen examples.

6.3 Classification Quality and ROC Analysis

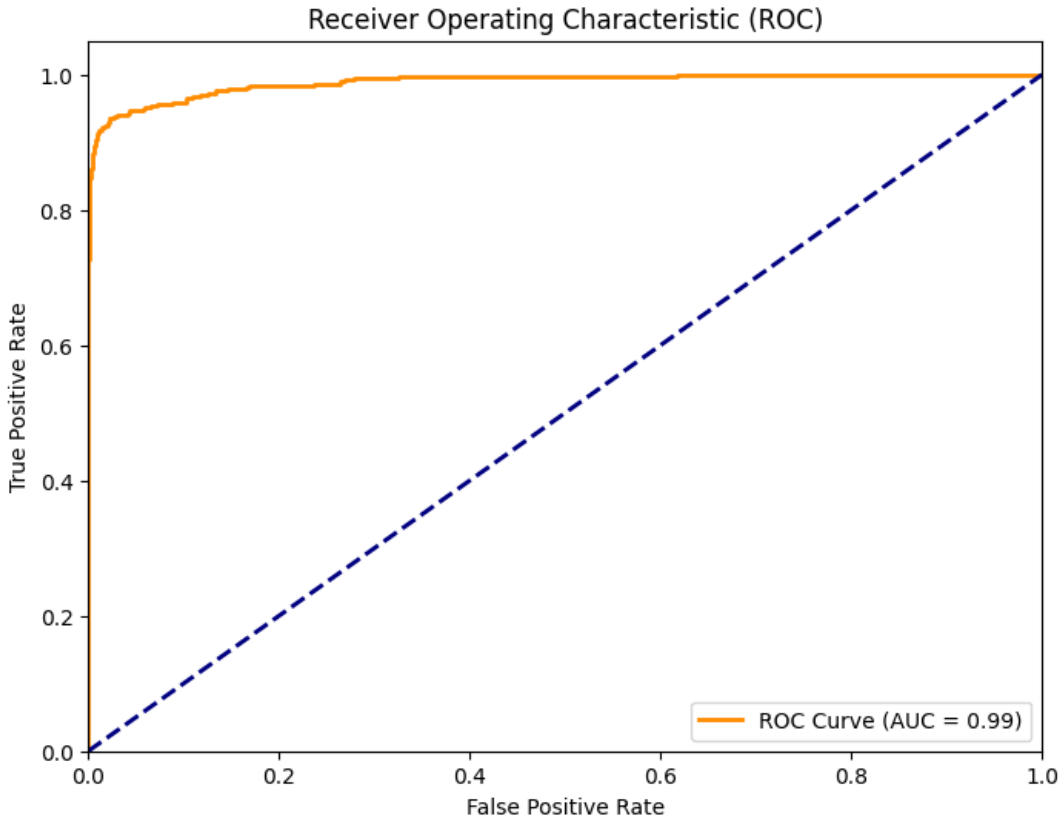

Fig 6.5 ROC Curve (AUC = 0.99)

Further confirming the model's classification strength, the ROC curve in Fig. 6.5 presented shows an AUC of 0.99. This virtually perfect score indicates the model's very high capability to differentiate between true positive and false positive instances, indicating high sensitivity and specificity.

6.4 Real-Time Deployment

To validate the practical applicability of the Adsift system, the model was deployed in a real-time environment using a fully functional FM Web Interface. The client-server architecture was developed using FastAPI for backend processing and React.js for the frontend interface, ensuring a seamless, responsive experience for users. The web application enables users to interactively monitor FM station status, observe real-time classification results, and witness automatic station switching whenever an advertisement is detected. Audio streams from multiple FM channels are continuously processed every five seconds, classified using the ResNet-18 model, and the results are dynamically updated on the interface.

Key functionalities demonstrated during real-time deployment include:

- **Live FM Streaming:** Users can listen to FM stations directly through the web interface.
- **Real-Time Classification Display:** The system shows whether the current segment is classified as “Music” or “Advertisement.”
- **Automatic Station Switching:** When an advertisement is detected, the system intelligently switches to a different active FM station, minimizing user disruption.
- **User Authentication:** Secure login and signup features ensure that only registered users can access the classification dashboard.
- **Station Status Monitoring:** A text file presents the list of available stations, their current classification status in Serial Monitor.

The following screenshots illustrate various aspects of the web application:

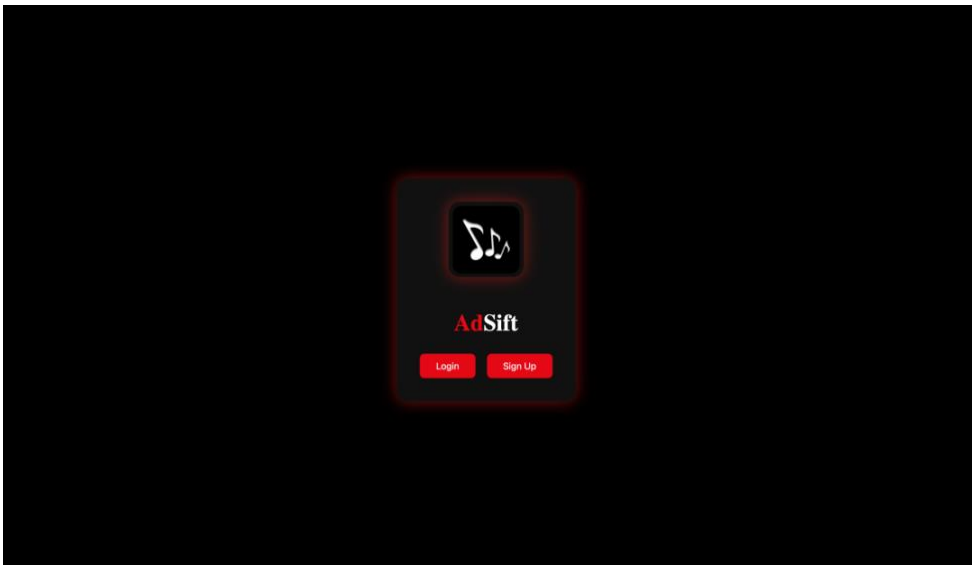

Fig 6.6 Welcome Page

The screenshot (Fig. 6.6) showcases the welcome page of the Adsift system, featuring the system's logo and clear options to either signup or login.

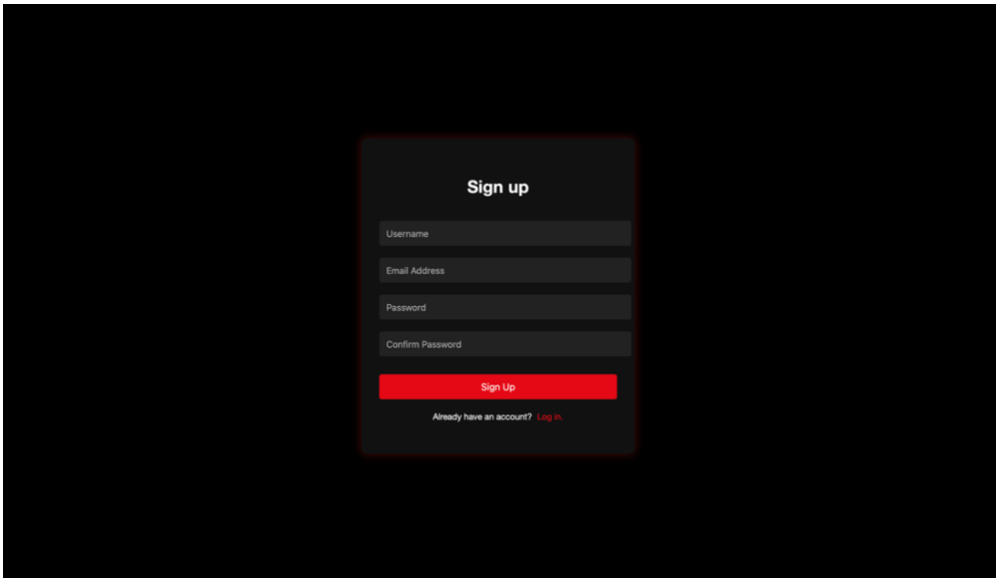

Fig 6.7 Signup Page

The screenshot (Fig. 6.7) showcases the signup page allows new users to create an account to access the FM classification features. The page is simple, clean, and intuitive, ensuring a smooth onboarding process.

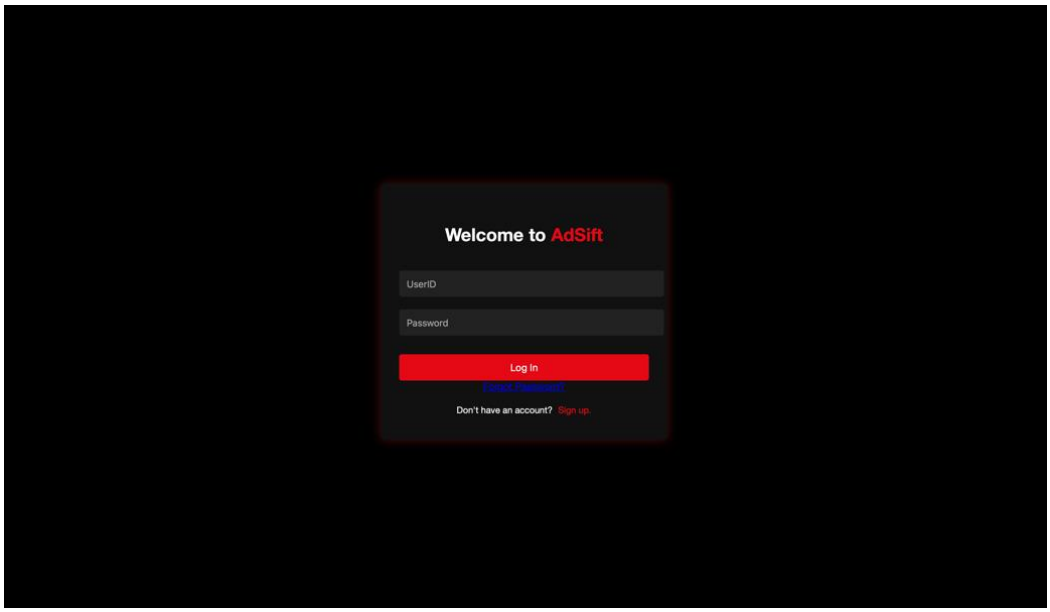

Fig 6.8 Login Page

The screenshot (Fig. 6.8) shows the login page enables users to securely log in to their accounts. Authentication is required to access the dashboard and real-time FM streaming features.

Fig 6.9 Adsift Music Player Page with Sidebar Open

The screenshot (Fig. 6.9) demonstrates the music player interface with the sidebar open, providing detailed information about the station, the classification results, and historical switching data.

Fig 6.10 Adsift Music Player Page with Sidebar Closed

The screenshot (Fig. 6.10) shows the music player interface with the sidebar closed. It provides an overview of the active FM station, the current classification status, and a playback interface for the user.

Overall, the real-time deployment of Adsift demonstrates its potential to enhance user experience by minimizing advertisement interruptions and delivering uninterrupted music playback, particularly in vehicular environments where driver focus and safety are critical.

CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1 Conclusion

The Adsift system successfully delivers a real-time, intelligent FM audio classification solution that enhances the listening experience by minimizing advertisement interruptions. By leveraging a fine-tuned ResNet-18 deep learning model on mel-spectrogram inputs, the system achieved outstanding performance metrics, with an accuracy of 97.45% under the 80-20 dataset split. Real-time deployment through a FastAPI-based ASGI server and a web application interface further demonstrated the system's practical viability.

Through the integration of multi-threading, GPU acceleration, and optimized data pipelines, Adsift ensures low-latency classification and seamless FM station switching even under high-traffic scenarios. The system's scalability, modular architecture, and REST API readiness position it as a future-proof platform adaptable to various domains beyond FM radio, including mobile, cloud, and automotive applications.

Overall, Adsift represents a significant step toward safer and more enjoyable FM listening experiences by intelligently filtering advertisements and maintaining uninterrupted music playback in real-time environments.

7.2 Future Work

While Adsift has demonstrated strong real-world performance, several enhancements are planned for future versions:

- **Handling Overlapping Audio**: Implementing advanced preprocessing techniques such as source separation and noise reduction to better classify overlapping advertisements and music segments.
- **Multi-Class Classification**: Extending beyond binary classification (ads and music) to detect additional categories such as news, podcasts, and talk shows.
- **Enhanced Real-Time Efficiency**: Further optimizing batch inference and multi-threading strategies to improve real-time performance and support a larger number of simultaneous FM streams.
- **Edge Computing Deployment**: Adapting the system to function efficiently on low-power devices like in-car infotainment systems, standalone FM receivers, and IoT audio modules.
- **API Expansion**: Enhancing REST API capabilities for easier third-party integration with mobile apps, smart assistants, and streaming platforms, making Adsift a versatile tool for intelligent audio filtering across various industries.

By addressing these future directions, Adsift aims to evolve into a comprehensive, scalable, and intelligent audio classification platform for a wide range of applications.

REFERENCES

1. M. Ghojazadeh, M. Farhoudi, M. Rezaei, S. Rahnemayan, M. Narimani, and H. S. Bazargani, "Effect of music on driving performance and physiological and psychological indicators: A systematic review and meta-analysis study," *Health Promot. Perspect.*, vol. 13, no. 4, p. 267, 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10790125/

2. A. B. Ünal, L. Steg, and K. Epstude, "The influence of music on mental effort and driving performance," *Accid. Anal. Prev.*, vol. 48, pp. 271–278, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S000145751200036X

3. Z. N. Jimison, "The effect of music familiarity on driving: A simulated study of the impact of music familiarity under different driving conditions," Master's thesis, Univ. of North Florida, 2014. [Online]. Available: https://digitalcommons.unf.edu/etd/539/

4. A. Kumar, B. Karan, S. S. Solanki, M. Chandra, and D. K. Singh, "Automatic tuning of radio stations based on listener's preference using software defined radio and MATLAB," *Eng. Appl. Artif. Intell.*, vol. 137, p. 109117, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0952197624012752

5. J. Álvarez et al., "Radia–radio advertisement detection with intelligent analytics," *arXiv preprint*, 2024. [Online]. Available: https://arxiv.org/abs/2403.03538

6. S. Venkatesh et al., "Artificially synthesising data for audio classification and segmentation to improve speech and music detection in radio broadcast," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2021, pp. 636–640. [Online]. Available: https://ieeexplore.ieee.org/document/9413784

7. C. Yang, X. Gan, A. Peng, and X. Yuan, "ResNet based on multi-feature attention mechanism for sound classification in noisy environments," *Sustainability*, vol. 15, no. 14, p. 10762, 2023. [Online]. Available: https://www.mdpi.com/2071-1050/15/14/10762

8. S. Hershey et al., "CNN architectures for large-scale audio classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2017, pp. 131–135. [Online]. Available: https://ieeexplore.ieee.org/document/7952132

9. M. S. Hussain and M. A. Haque, "SwishNet: A fast convolutional neural network for speech, music, and noise classification and segmentation," *ICTer*, vol. 12, no. 2, pp. 17–23, 2019. [Online]. Available: https://icter.sljol.info/ Also available: *arXiv preprint*, 2018. [Online]. Available: https://arxiv.org/abs/1812.00149

10. K. Zaman, M. Sah, C. Direkoglu, and M. Unoki, "A survey of audio classification using deep learning," *IEEE Access*, vol. 11, pp. 106620–106649, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10280412

11. J. Zhang, “Music genre classification with ResNet and Bi-GRU using visual spectrograms,” *arXiv preprint*, 2023. [Online]. Available: https://arxiv.org/abs/2307.10773

12. E. Tsalera, A. Papadakis, and M. Samarakou, “Comparison of pretrained CNNs for audio classification using transfer learning,” *J. Sens. Actuator Netw.*, vol. 10, no. 4, p. 72, 2021. [Online]. Available: https://www.mdpi.com/2224-2708/10/4/72

13. S. Dieleman, P. Brakel, and B. Schrauwen, “Audio-based music classification with a pretrained convolutional network,” in *Proc. 12th Int. Soc. Music Inf. Retrieval Conf. (ISMIR)*, 2011, pp. 669–674. [Online]. Available: https://ismir2011.ismir.net/papers/PS6-13.pdf

14. G. Karunarathna, L. Jayaratne, and K. Gunawardana, “Classification of voice content in the context of public radio broadcasting,” *Int. J. Adv. ICT Emerging Reg.*, vol. 12, no. 2, pp. 17–23, 2019. [Online]. Available: https://icter.sljol.info/articles/abstract/10.4038/icter.v12i2.7249/

15. S. Konduri, K. V. Pendyala, and V. S. Pendyala, “Kritisamhita: A machine learning dataset of south Indian classical music audio clips with tonic classification,” *Data Brief*, vol. 55, p. 110730, 2024.[Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340923012622

16. R. Müller-Cajar, “Detecting advertising in radio using machine learning,” BSc thesis, Univ. of Auckland,2007.[Online].Available:https://www.cs.auckland.ac.nz/courses/compsci705s1c/archive/2007/assignments/ReneMullerCajar.pdf

17. R. Kotsakis and C. Dimoulas, “Extending radio broadcasting semantics through adaptive audio segmentation automations,” *Knowledge*, vol. 2, no. 3, pp. 347–364, 2022. [Online]. Available: https://www.mdpi.com/2673-9585/2/3/21

.