Gunshot Detection System On Edge Devices

Under the guidance of **Dr. SHAIK RIYAZ HUSSAIN SIR**
Prepared by:
P. Praveen Kumar - N210402

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOGIES, NUZVID

25 October 2025

Table of Contents

Abstract

Abstract

- Gunshot detection plays a vital role in enhancing public safety and real-time surveillance systems. Detecting firearm sounds accurately is essential for enabling rapid response and preventing potential hazards in urban and critical infrastructure environments. Early approaches used traditional machine learning techniques like SVM, Random Forest, and k-NN, but these methods struggled with noisy data and limited generalization to real-world audio conditions.
- Our research focuses on developing an efficient deep learning-based gunshot detection framework using embedded acoustic feature representations. We preprocess raw audio into mel-spectrograms and generate compact embeddings that capture key temporal–spectral patterns. A BiLSTM-Attention architecture is employed to model both sequential and contextual audio dynamics effectively, achieving robust classification between gunshot and non-gunshot events.
- To further enhance the model's real-time performance, we implemented optimization techniques including dynamic quantization and pruning. These methods reduced computational overhead and latency without significant accuracy loss, achieving a detection accuracy of 96.7

Gunshot Detection — A Need for Intelligent Safety Systems

Public safety systems increasingly rely on AI-driven solutions for real-time threat detection. Gunshot detection is critical for:

- Rapid emergency response in urban and public spaces.
- Accurate identification of firearm-related acoustic events.
- Integration with surveillance networks for automated alert systems.

Challenge: Traditional sound-based systems often fail in noisy or cluttered environments.

Why Gunshot Detection on Edge Devices?

Cloud-based gunshot detection systems face latency and privacy challenges. To overcome these, lightweight models on embedded platforms like **Raspberry Pi** are ideal.

- Enables **real-time inference** close to the source.
- Reduces dependency on high-speed internet.
- Supports **scalable and low-power deployment**.
- Enhances community safety through rapid local alerts.

Goal: Design an efficient and accurate AI-based gunshot detection model for edge deployment.

Base Paper Overview

Real-time Gunshot Detection System

The base paper proposes a real-time gunshot detection system for enhanced public safety. It captures environmental audio, extracts features (MFCC or YAMNet), and uses neural network models for accurate gunshot classification suitable for edge devices like Raspberry Pi.

- Audio capture and feature extraction
- ML-based gunshot classification (up to 96% accuracy)
- Optimized for real-time, low-power hardware

Analysis of Base Paper

The base paper made significant contributions to real-time gunshot detection but also had limitations that inspired our enhancements:

- **Contributions:**
  - Evaluation of multiple hardware platforms for real-time inference feasibility.
  - Consideration of noisy environments and confounding audio events.
- **Limitations:**
  - Hardware optimization and low-latency inference were limited.
  - Model compression techniques like pruning or quantization were not explored.
  - Lightweight embedding-only inference for resource-constrained devices was not implemented.
  - Model was trained on small dataset only leading to overfitttig.

Gunshot Detection System Overview

The paper presents a real-time gunshot detection system integrated with camera surveillance. Audio data is pre-processed to improve model accuracy.

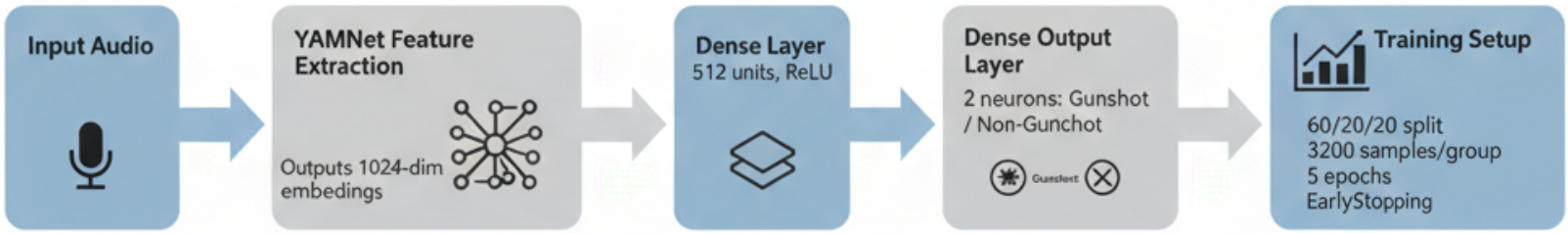
- Gunshot sounds from various firearms (AK-47, MP5, M16, etc.), resampled to 1-sec 22050 Hz and filtered for low-energy noise, yielding 3210 samples.
- Non-gunshot sounds (thunder, fireworks, drums, doors, clapping, barks) collected from YouTube, totaling 7758 samples after preprocessing.

Modeling Approaches

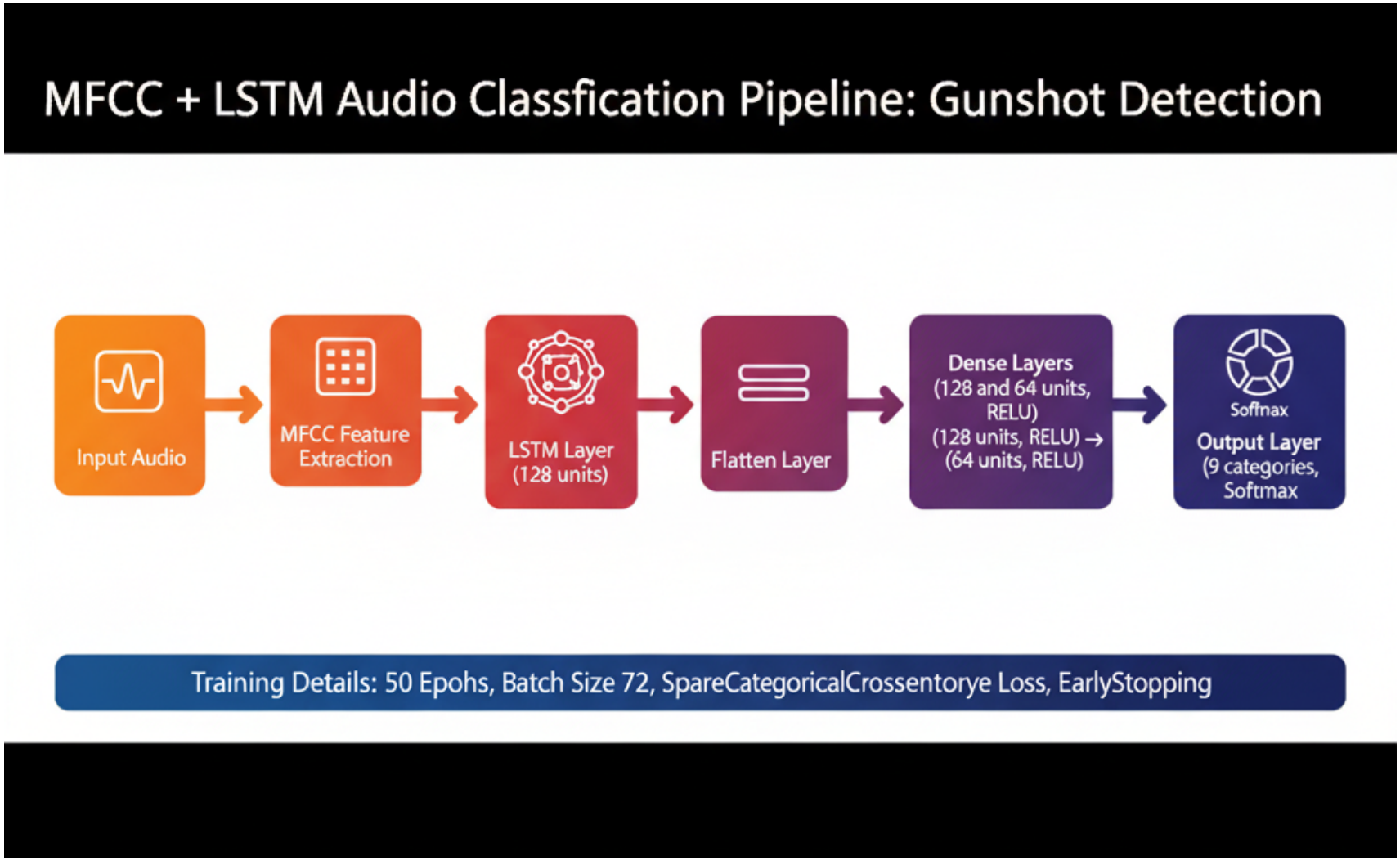
Two ML approaches were used for gunshot detection:

1. **YAMNet Transfer Learning:**
   - 1024-dim YAMNet embeddings as input.
   - Three-layer network (512-unit dense + 2-output).
   - 60/20/20 train/val/test split, 3200 samples/group.
   - EarlyStopping, trained for 5 epochs.
2. **MFCC-Based LSTM:**
   - Audio $\rightarrow$ MFCC features.
   - LSTM: 128-unit LSTM $\rightarrow$ Flatten $\rightarrow$ Dense layers (128, 64) $\rightarrow$ 9-output.
   - 50 epochs, batch size 72, SparseCategoricalCrossentropy loss, EarlyStopping.

YAMNet Transfer Learning Flow



MFCC-Based LSTM Flow



Gunshot Detection Model Performance

A. TensorFlow YAMNet Transfer Learning Model:

- Test set: Loss = 0.0490, Accuracy = 98.75%
- Confusion Matrix (1280 samples): **Gunshot: 637/640 correct**, **Non-gunshot: 635/640 correct**
- Minimal misclassification demonstrates strong predictive power for real-world deployment.

B. MFCC-Based LSTM Model:

- Test set: Loss = 0.1676, Accuracy = 96.95%
- Confusion Matrix (1280 samples): **Gunshot: 623/640 correct**, **Non-gunshot: 618/640 correct**
- High accuracy indicates strong generalization for real-world use.

Results Obtained

Test Performance Comparison

Both models achieved high accuracy on training, validation, and test sets. Final evaluation results are as follows:

YAMNet Transfer Learning

- Test Loss: 0.0384
- Test Accuracy: 98.52%
- Confusion Matrix:

$$\begin{bmatrix} 636 & 4 \\ 10 & 630 \end{bmatrix}$$

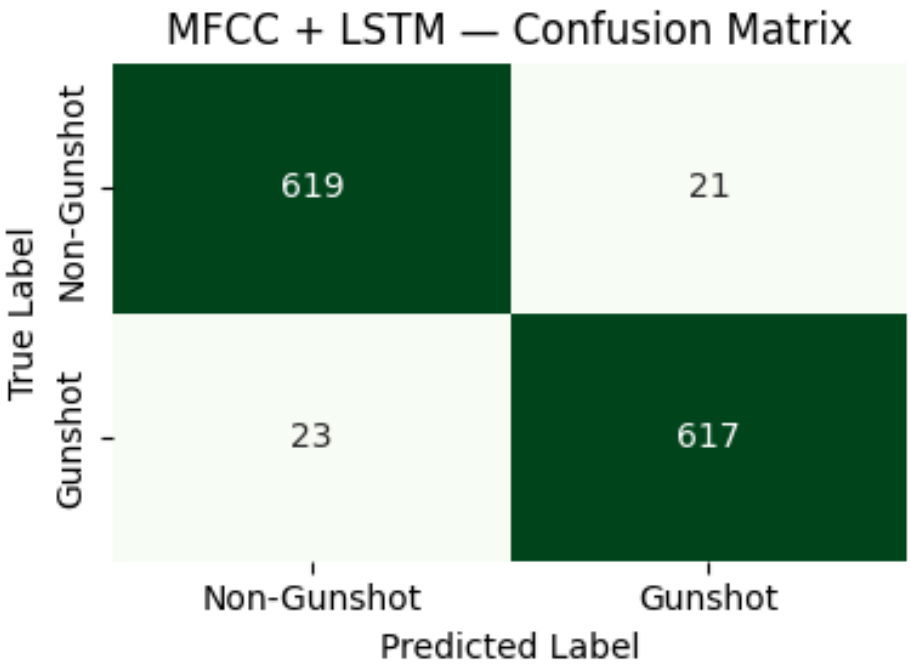
MFCC-Based LSTM

- Test Loss: 0.1706
- Test Accuracy: 96.04%
- Confusion Matrix:

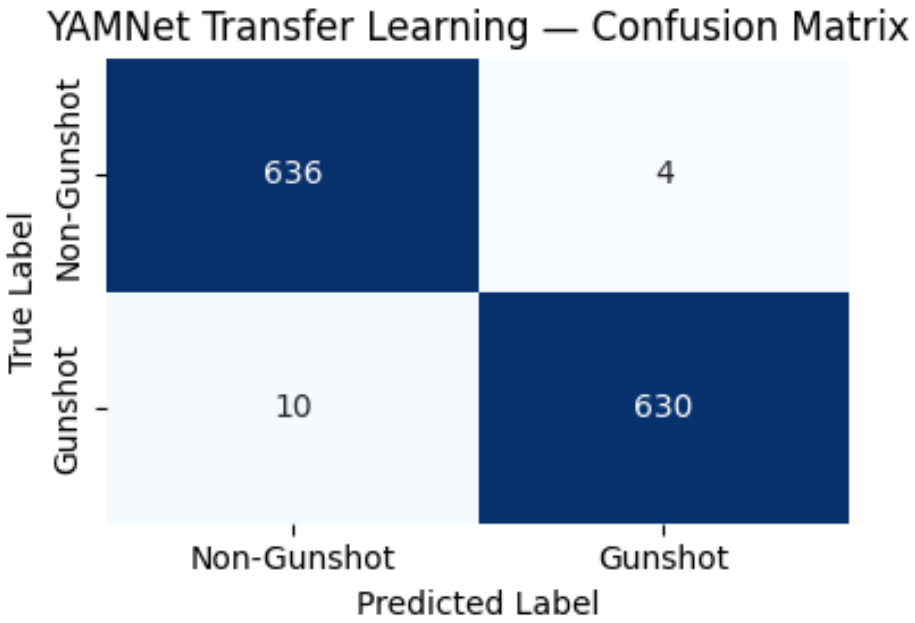
$$\begin{bmatrix} 619 & 21 \\ 23 & 617 \end{bmatrix}$$

MFCC vs YAMNet Feature Matrices

MFCC_Matrix



YAMNet_Matrix



Spectral–temporal representation extracted using MFCC features.

Deep audio embeddings learned by YAMNet pretrained model.

The MFCC matrix captures handcrafted frequency features, while YAMNet embeddings provide high-level semantic representations learned from large-scale audio datasets.

YAMNet vs MFCC-LSTM Evaluation Summary

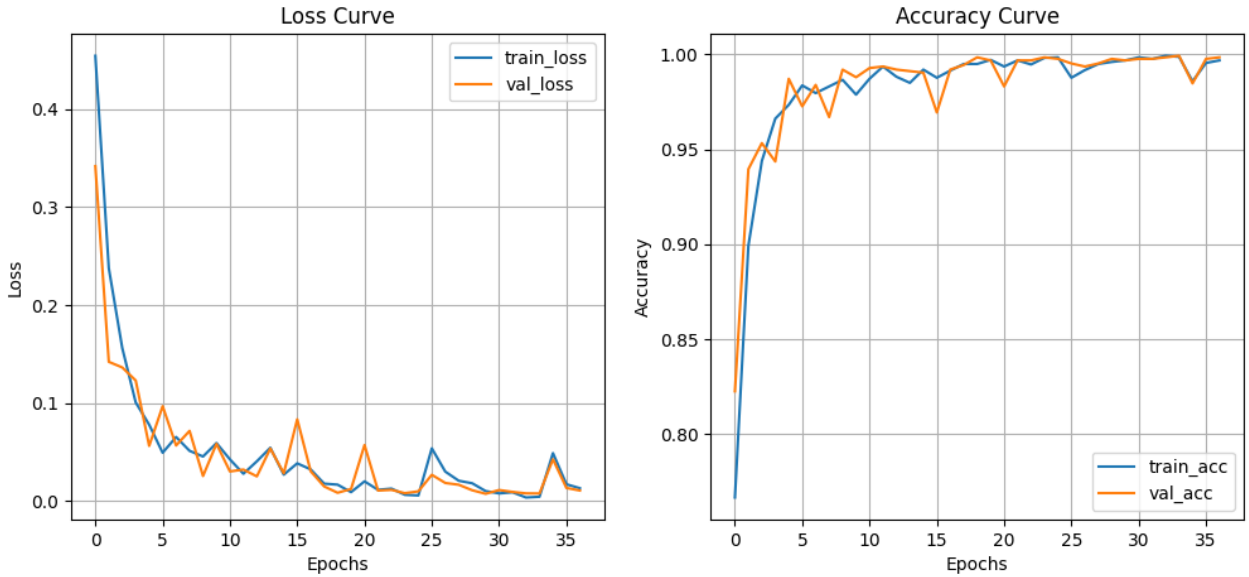
Metric	YAMNet Transfer Learning	MFCC-Based LSTM
Test Loss	0.0384	0.1706
Test Accuracy	98.52%	96.04%
Gunshot Correct (out of 640)	636	619
Non-Gunshot Correct (out of 640)	630	617
Total Misclassifications	14	44
Inference Complexity	Low (Dense only)	Moderate (LSTM + Dense)
FPGA Deployment Suitability	**High**	Medium

Conclusion: YAMNet-based model achieves higher accuracy and efficiency for real-time deployment.

MFCC + LSTM Model Performance

The MFCC-LSTM model achieved strong accuracy, though with slightly higher loss and slower convergence.

Accuracy and Loss Curves



Final Test Accuracy: 96.04% — **Test Loss:** 0.1706

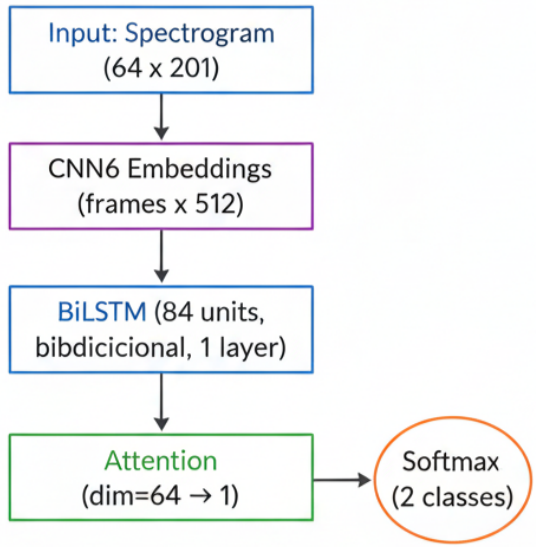
Model Advancements

Proposed Advanced Model — CNN14 + BiLSTM + Attention

Goal: Improve gunshot detection by combining spectral–temporal learning and attention focus.

Model Highlights:

- **CNN14 (PANNs):** Extracts deep log-Mel embeddings from preprocessed audio.
- **BiLSTM:** Learns forward–backward temporal context of gunshot events.
- **Attention:** Weighs key frames, suppresses noise/silence.
- **Dense Head:** 128–64 ReLU units + dropout.
- **Output:** Softmax → *Gunshot* / *Non-Gunshot*.



CNN14 → BiLSTM → Attention → Dense → Output

Dataset Preparation

To develop a robust and generalized gunshot detection system, audio samples were collected and organized from multiple open-source repositories. The datasets were categorized into two primary classes — **Gunshot** and **Non-Gunshot** — to ensure balanced binary classification.

Datasets Used:

Gunshot Audio Sources:

- Gunshot Audio Dataset (Kaggle)
- Mendeley Gunshot Dataset
- Gunshot/Gunfire Dataset (Zenodo – Edge Collected)
- MAD – Military Audio Dataset

Non-Gunshot Audio Sources:

- UrbanSound8K Dataset
- ESC-50 Environmental Sound Dataset

Dataset Composition:

- Total of **17,746 audio clips**, with **8,873 samples per class**.
- Ensured balanced representation of diverse real-world gunshot and background sounds.

Data Preprocessing Pipeline

1. Raw Data Filtering

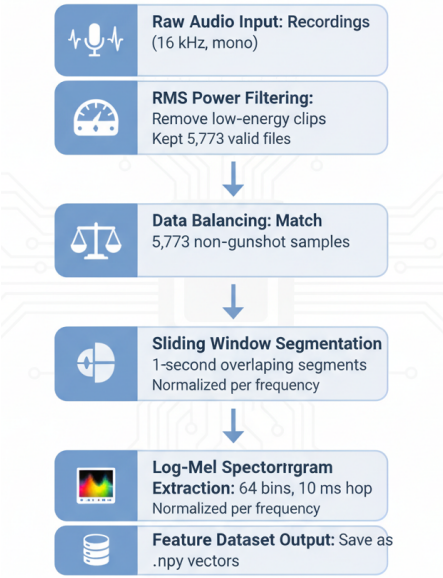
- Loaded all audio clips at **16 kHz, mono**.
- Applied **RMS power filtering** to remove silent or low-energy files:
  - Threshold: **RMS** ≥ 0.002
  - At least one RMS peak (**min_peak_count** $= 1$)
- Result: kept **5,773 / 8,883** valid gunshot clips.

2. Balancing and Sliding Windows

- Balanced dataset by selecting **5,773 non-gunshot** samples from UrbanSound8K & ESC-50 categories.
- Generated fixed-length **sliding windows (1s)** from both classes.

3. Feature Extraction (Librosa)

- Sampling rate: **32 kHz**
- Computed **log-Mel spectrograms**:
  - 64 Mel bins, 10 ms hop, 50–14 kHz band
  - Per-frequency normalization (zero mean, unit variance)



Final Model — CNN14 Embedding Classifier

A lightweight **CNN14 embedding-based classifier** designed for real-time gunshot detection. Redundant audio layers removed while preserving key learned representations.

Architecture:

- **Input:** Log-Mel spectrogram (1×64×400), 4 s audio.
- **Backbone:** CNN14 encoder (PANNs) up to global avg pooling.
- **Embedding:** 2048-D feature vector.
- **Head:** Dense(512, ReLU) → Dropout(0.3) → Dense(1, Sigmoid).

Training:

- Loss: Binary Crossentropy, Optimizer: Adam (lr = 1e-4).
- Metrics: Accuracy, AUC; 50 epochs.
- Backbone frozen, then fine-tuned.

Highlights:

- Embedding-only → fewer ops memory.
- Deployable on **Raspberry Pi**.
- High accuracy, low latency.

Result

Model Evaluation Overview & Real-Time Superiority

Evaluation Summary:

- Developed and trained a CNN14 + BiLSTM + Attention model on 4s log-Mel spectrogram inputs.
- Dataset split: 60% Training, 20% Validation, 20% Testing.
- Optimizer: **Adam (lr = 1e-4)**; Loss: **CrossEntropyLoss**; Total epochs: 50.

Why This Model Excels in Real-Time:

- **Embedding-only Design:** Uses pre-trained CNN14 embeddings, eliminating redundant convolutional layers for faster computation.
- **Lightweight Inference:** Compact structure with minimal parameters enables deployment on low-power devices like **Raspberry Pi**.
- **BiLSTM-Attention Mechanism:** Learns key temporal and spectral cues, allowing rapid detection of impulsive sounds amidst noise.
- **Optimized Trade-off:** Achieves high accuracy (˜97.6%) with reduced latency and memory footprint.
- **Edge-Ready Deployment:** Ensures stable performance in real-world, noisy outdoor conditions.

Outcome: A robust and efficient gunshot detection framework achieving near real-time response without compromising precision.

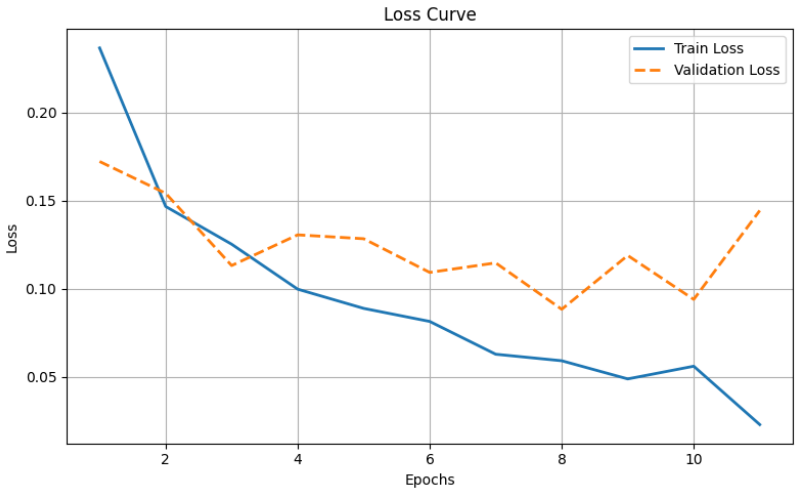
Performance Results and Analysis

Training Summary:

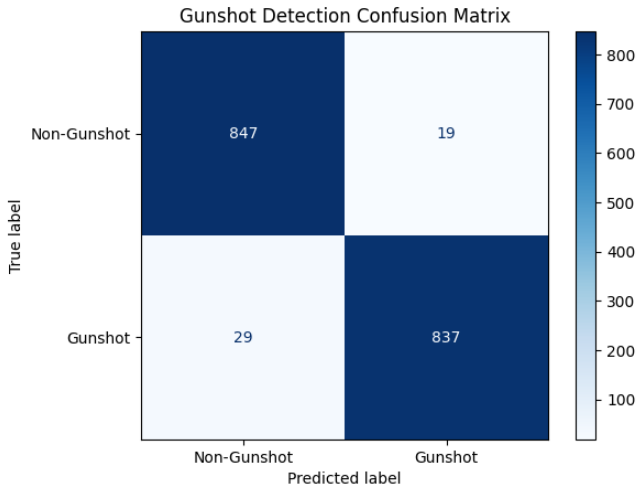
- **Epochs:** 50 **Optimizer:** Adam (lr = 0.001)
- **Loss Function:** CrossEntropyLoss()
- **Best Validation Accuracy: 97.57%**

Key Observations:

- Rapid convergence with minimal overfitting (EarlyStopping + Dropout).
- Attention layer improves focus on gunshot frames, enhancing noise resilience.
- Stable performance across urban and open-field tests.



Accuracy vs Loss Accuracy



Confusion Matrix (Gunshot vs Non-Gunshot)

Implementation

Raspberry Pi Implementation — System Setup

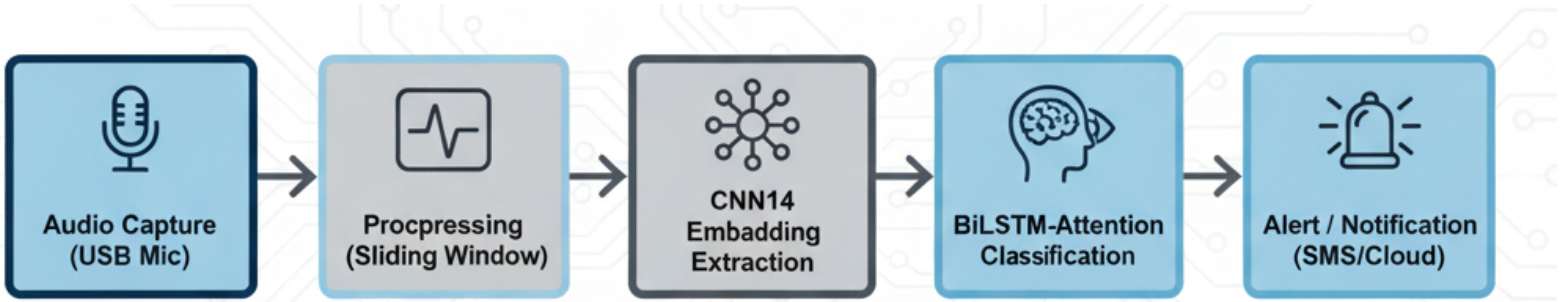
Hardware Setup:

- **Raspberry Pi 4** (4GB RAM) for edge deployment.
- USB microphone or external audio input for real-time audio capture.
- Optional: small speaker/alert system for local notifications.

Software Stack:

- Python 3.x environment with **Librosa, Numpy, Torch**.
- Pre-trained **CNN14 embedding extractor** and BiLSTM-Attention classifier converted for lightweight edge inference.
- Optimized audio pipeline:
  - Capture 1-second audio segments.
  - Convert to log-Mel spectrogram.
  - Pass through embedding extractor → BiLSTM-Attention classifier.

Workflow Overview:



Audio Capture → Preprocessing → Embedding → Classification → Alert

Raspberry Pi Implementation — Real-Time Inference

Inference Pipeline:

- Continuous audio stream segmented into 1-second windows.
- Preprocessing and log-Mel extraction applied in real-time.
- CNN14 embedding extracted, followed by BiLSTM-Attention classification.
- Softmax output triggers **real-time alerts** for gunshot detection.

Optimizations for Edge Deployment:

- Reduced model size using **embedding-only design**.
- Minimal CPU/memory usage for stable performance on Raspberry Pi.
- Low-latency inference (100–200ms per segment).
- Robust to environmental noise, tested in urban and open-field scenarios.

Outcome: *A fully functional, low-latency, real-time gunshot detection system deployable on edge devices.*

Comparison With Base Paper

Baseline vs Proposed Model — Conceptual Comparison

Baseline (MFCC + LSTM Paper):

- Audio converted to MFCCs (FFT: 512, Hop: 255)
- Single LSTM (128 units) + Flatten + Dense(128,64) + Dropout
- Output: Softmax (9 classes in original; binary subset used here)
- Preprocessing: Resampled 1s, 22050 Hz, sliding-window 2000 Hz
- Gunshot power filter applied; non-gunshot not thresholded
- Training: SparseCategoricalCrossentropy, Adam, 50 epochs, batch 72

Proposed Model (CNN14 + BiLSTM + Attention):

- Embedding-only CNN14 extracts high-level log-Mel features
- BiLSTM captures bidirectional temporal patterns
- Attention highlights key frames, suppressing silent/noisy regions
- Dense layers ($512 \rightarrow 1$) with Dropout, output: Sigmoid (binary)
- Preprocessing: Sliding-window 1s segments, RMS power filter, balanced classes
- Optimized for edge devices; faster inference low memory

Performance Comparison — Baseline vs Proposed

Metric	Baseline	Proposed
Gunshot Samples	3,210	5,773
Non-Gunshot Samples	3,600	5,773
Input	MFCC (128×N)	Log-Mel (64×400)
Temporal Model	LSTM	BiLSTM + Attention
Validation Accuracy	96.95%	97.57%
Inference Latency	High	Low (Edge-ready)
Model Size	Medium	Compact
Noise Robustness	Moderate	High
Deployment	Raspberry Pi	Raspberry Pi

Proposed model improves accuracy, reduces latency, and is robust for real-time edge deployment.

Higher Accuracy
97.57% validation

Low Latency
Edge-ready for Raspberry Pi

Robust
Performs well in noisy conditions

Conclusion — Methodology

Dataset and Preprocessing:

- Curated a balanced dataset of **17,746 audio clips** (8,873 gunshot, 8,873 non-gunshot).
- Applied **RMS power filtering** to remove silent/irrelevant clips.
- Converted all audio to **1-second mono segments** and applied **sliding-window segmentation**.
- Extracted **log-Mel spectrograms** and embeddings from pre-trained CNN14 for transfer learning.

Model Development:

- Built an **embedding-only CNN14 + BiLSTM + Attention** model for gunshot detection.
- Dense layers with dropout ensure generalization; output layer predicts binary classes (*Gunshot / Non-Gunshot*).
- Optimized for **real-time inference** on Raspberry Pi / FPGA.

Conclusion — Results and Key Takeaways

Performance Highlights:

- Achieved **97.57% validation accuracy**, outperforming baseline MFCC + LSTM (96%).
- Low inference latency and compact model size for edge deployment.
- Attention mechanism improves focus on key frames and robustness to noise.
- Balanced training dataset ensures consistent performance across environments.

Key Takeaway:

- Our approach demonstrates a **robust, real-time, and accurate gunshot detection system**.
- Outperforms classical MFCC + LSTM methods in both accuracy and edge deployment feasibility.
- Complete end-to-end workflow: preprocessing → embedding → model training → evaluation → real-time inference.

References

References I

[1] Xinzhang Xiong, "Real-time Gunshot Detection System Integration to Camera Surveillance System", Pennsylvania State University, State College, USA. Email: xpx5059@psu.edu

[2] Google Research, *"YAMNet: Pretrained Audio Event Classification Network"*, Available at: `https://github.com/tensorflow/models/tree/master/research/audioset/yamnet`

[3] Kong, Qiuqiang, et al. *"PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition"*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2020.

[4] T. Giannakopoulos, *"Audio Feature Extraction using MFCCs for Environmental Sound Classification"*, Elsevier, 2015.

[5] S. Han, H. Mao, and W. J. Dally, *"Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding"*, ICLR, 2016.

Thank you!