ENVIORNMENTAL AUDIO CLASSIFICATION USING ATTENTION-BASED DEEP LEARNING MODELS

A Thesis Submitted
in Partial Fulfillment of the Requirements for the
Degree Of

MASTER OF TECHNOLOGY

in

CONTROL & INSTRUMENTATION

by

Yash Kumar
23/C&I/03

Under the Supervision of
Mr. Rohan Pillai
Mr. Gaurav Kaushik

Department of Electrical Engineering
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, New Delhi-110042

MAY, 2025

ACKNOWLEDGEMENT

I would like to thank **Dr. Rachna Garg**, the Head of Department, Department of Electrical Engineering, for their continuous support and for providing a conducive environment for research. Their leadership and encouragement have been crucial in facilitating the progress of my work.

I take immense pleasure to express my deep and sincere gratitude to my esteemed supervisor, **Mr. Rohan Pillai**, for his invaluable guidance and spending of his precious hours for my work. His excellent cooperation and suggestion through stimulating and beneficial discussions provided me with an impetus to work and made the completion of the thesis possible. I also extend my sincere thanks to my co-supervisor **Mr. Gaurav Kaushik** for offering invaluable assistance and helping me in my research work which boost up my confidence and pave the way to plan for surveying at the initial stages. He provided me constant encouragement and support in all possible ways.

I also appreciate and heartily thanks to all faculty member of **Department of Electrical Engineering** and all my lab mates for their constant support in my research work. I am also thankful to all my classmates and technical staff for supporting and helping me throughout the research work.

Finally, yet importantly, I would like to pay high regards to lord almighty, my Parents, my family members and my friends for their inspiration and motivation throughout my work and lifting me uphill this phase of life.

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Bawana Road, New Delhi-110042

CANDIDATE'S DECLARATION

I, **Yash Kumar (23/C&I/03)** student of M.Tech (Control & Instrumentation), hereby certify that the work which is being presented in the thesis entitled "**ENVIORNMENTAL AUDIO CLASSIFICATION USING ATTENTION-BASED DEEP LEARNING MODELS**" in partial fulfillment of the requirements for the degree of Master of Technology, submitted in Department of **Electrical Engineering**, Delhi Technological University, New Delhi is an authentic record of my own work and not copied from any source without proper citation which is carried out under the supervision of **Mr. Rohan Pillai** and co-supervision of **Mr. Gaurav Kaushik**.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Candidate's Signature

This is to certify that the student has incorporated all the corrections suggested by the examiners in the thesis and the statement made by the candidate is correct to the best of our knowledge.

Signature of Supervisors **Signature of External Examiner**

Date: May 2025

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Bawana Road, New Delhi-110042

CERTIFICATE BY THE SUPERVISOR

Certified that **Yash Kumar (23/C&I/03)** has carried out their research work presented in this thesis entitled "**Enviornmental Audio Classification Using Attention-Based Deep Learning Models**" for the award of Master of Technology from Department of Electrical Engineering, Delhi Technological University, New Delhi under our supervision. The thesis embodies results of original work, and studies carried out by the students himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Mr. Rohan Pillai
Department of Electrical Engineering
Delhi Technological University

Mr. Gaurav Kaushik
Department of Electrical Engineering
Delhi Technological University

Date: May 2025

ABSTARCT

Environmental audio classification involves the identification and categorization of real-world sound events based on their acoustic characteristics. Unlike digital classification tasks that rely on static features, audio classification requires models capable of capturing both temporal and spectral dynamics of non-stationary signals. This study presents a deep learning-based approach that utilizes time-frequency representations of environmental audio, specifically Mel-spectrograms, as input to a deep learning architecture enhanced with attention mechanisms. The raw audio data underwent pre-processing steps including resampling and conversion into Mel-spectrograms to extract meaningful time-frequency features suitable for model training. These mechanisms allow the model to selectively focus on relevant temporal features, improving its ability to differentiate between overlapping or acoustically similar events. Comparative evaluation with traditional convolutional neural networks (1D CNNs) highlights the advantages of attention-based architectures in modelling long-range dependencies and capturing richer contextual information from audio sequences. The audio pre-processing pipeline, model design, and evaluation procedures are implemented using Python-based libraries and advanced deep learning frameworks. The proposed system demonstrates robustness in classifying a variety of sound types, showing potential for deployment in real-time monitoring applications such as smart surveillance, public safety systems, and ambient sound analysis in urban environments. Experimental results show that attention-based models offer improved classification performance and adaptability compared to conventional architectures, making them well-suited for complex acoustic environments. This work contributes to the growing field of intelligent acoustic sensing by offering a flexible and efficient model architecture that adapts to complex audio patterns.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 OVERVIEW

Environmental Sound Classification (ESC) involves detecting and labeling everyday sounds—like barking dogs, car horns, drilling, or children playing—based on their unique acoustic properties. Unlike more structured audio signals such as speech or music, environmental sounds are often irregular, overlapping, and vary in both length and volume, making them difficult to classify accurately.

ESC plays a critical role across multiple sectors. It's used in smart surveillance, public safety alerts, wildlife monitoring, autonomous navigation, and assistive tools for individuals with hearing impairments. As IoT devices and intelligent systems become more widespread, the need for real-time and reliable sound classification has grown significantly.

Earlier ESC systems primarily depended on manually engineered features—like MFCCs, zero-crossing rate, and spectral roll-off—and used traditional machine learning algorithms such as SVM, KNN, or Random Forests. While these methods worked reasonably well in controlled conditions, they often fell short when applied to the unpredictability of real-world environments.

With the advent of deep learning, the landscape of ESC has evolved considerably. Convolutional Neural Networks (CNNs), in particular, have gained traction for their ability to automatically extract relevant patterns from Mel-spectrograms—a visual representation of audio signals in the frequency domain. CNNs excel at detecting localized features but often miss long-range dependencies in the audio, limiting their performance in more complex scenarios.

To overcome this, researchers have explored sequential models such as recurrent neural networks (RNNs), long short-term memory (LSTM) networks, and more recently, transformer-based architectures. Transformers utilize self-attention mechanisms, enabling them to attend to important parts of the audio sequence, regardless of their position in time. This is particularly beneficial in ESC tasks, where vital acoustic information can occur sporadically or be masked by background noise.

This thesis introduces a hybrid CNN-Transformer architecture designed to combine the strengths of both paradigms. The CNN layers act as front-end feature extractors to learn spatial features from Mel-spectrograms, while

Transformer layers capture long-term temporal dependencies through multi-head self-attention. The model is trained using the UrbanSound8K dataset, a widely used benchmark in ESC research, which contains 8732 annotated audio samples from 10 urban sound classes.
To enhance the model's generalizability and robustness, we implement SpecAugment—a data augmentation technique that masks sections of spectrograms in the time and frequency domains. Additionally, dropout and batch normalization are applied for regularization, and 10-fold cross-validation is used to evaluate the model comprehensively.
Overall, this study aims to develop a scalable, accurate, and computationally feasible system for environmental sound classification using attention-enhanced deep learning. By bridging CNNs and Transformers, we propose a robust framework that significantly improves performance over traditional and standalone models.

1.2 MOTIVATION

The primary motivation for this research stems from the limitations of existing ESC systems in accurately classifying complex, real-world audio environments. As urban spaces become increasingly connected through smart devices, the demand for scalable and accurate sound recognition systems has grown exponentially.

In real-world scenarios like smart surveillance or autonomous systems, it's not enough for a model to simply detect sound events—it also needs to understand when and how those sounds occur over time. While Convolutional Neural Networks (CNNs) are effective at identifying localized patterns in spectrograms, they don't inherently capture the broader temporal structure of audio. On the flip side, Transformers excel at interpreting sequences by attending to the entire input at once, providing a global perspective. However, they tend to overlook fine-grained, localized details that CNNs are good at picking up.

By integrating the strengths of both CNNs and Transformers into a hybrid architecture, the goal is to build a system that can not only recognize what sound occurred, but also grasp the surrounding temporal context with higher accuracy and reliability

- Efficiently process and analyze real-time audio streams
- Accurately classify overlapping and context-dependent sound events
- Generalize well across different environments and conditions

Additionally, incorporating techniques like SpecAugment and other forms of regularization helps the model handle noise and inconsistencies in real-world audio

more effectively. This improves the model's ability to generalize and remain stable across different environments. The hybrid design also brings advantages in terms of computational efficiency, making it a strong candidate for deployment on resource-constrained devices such as edge hardware or embedded systems.

To sum up, combining attention mechanisms with convolutional feature extraction provides a compelling approach to tackling the challenges of environmental sound classification. The proposed model is designed not only to deliver high accuracy but also to act as a solid stepping stone for future research and practical applications in smart acoustic sensing technologies

CHAPTER 2 LITERATURE REVIEW

Environmental Sound Classification (ESC) has become an important area of research within machine hearing and acoustic scene analysis. Over time, the field has evolved considerably, especially with the introduction of deep learning, attention-based models, and self-supervised learning techniques. These advancements have made it possible to move beyond manually crafted audio features and toward systems that learn directly from raw data. This section offers a broad overview of key studies in the ESC domain, with a particular focus on work involving the UrbanSound8K dataset and hybrid model architectures.

The foundation for much of the current ESC research was laid by Salamon et al. (2014), who developed the UrbanSound8K dataset. Their study combined MFCC features with random forest classifiers and also tested basic CNN models, establishing early benchmarks and highlighting the importance of consistent preprocessing and evaluation standards.

Following this, Piczak (2015) showed how 2D CNNs could be applied to log-Mel spectrograms, using stacked input and data augmentation to boost performance. This relatively simple model helped shape best practices that are still common today in ESC systems.

In 2017, Tokozume et al. challenged the reliance on spectrograms by proposing a model that could learn directly from raw audio waveforms. Their deep CNN framework demonstrated that meaningful features could be learned without time-frequency transformations, as long as there was enough training data.

Huzaifah (2017) further explored CNN configurations, experimenting with kernel sizes and network depth. His findings emphasized that model performance is highly sensitive to architecture design, especially when working with relatively small datasets like UrbanSound8K.

Wang et al. (2019) introduced attention mechanisms into CNN-based ESC models, allowing the network to focus on the most important parts of the input. This approach led to better performance and gave insights into which segments of audio the model found most relevant.

Kumar et al. (2021) presented a hybrid CNN-Transformer architecture, where CNN layers handled local feature extraction and Transformer blocks captured longer temporal dependencies. Their results showed marked improvements in classification accuracy.

Building on this idea, Chen et al. (2022) combined Transformer attention with SpecAugment, a data augmentation strategy that masks time and frequency components. Their model not only improved generalization but also performed better in noisy environments.

Gairola et al. (2021) trained a custom Transformer from scratch on the ESC-50 and UrbanSound8K datasets, using spectral positional encodings to help the model better understand temporal structure. This gave the Transformer an edge over traditional CNNs in modeling audio sequences.

Zhou et al. (2020) experimented with a CRNN-Transformer hybrid to handle polyphonic audio events, making it one of the first models in ESC to tackle multi-label classification in overlapping sound scenarios.

Gong et al. (2021) introduced the Audio Spectrogram Transformer (AST), adapting the Vision Transformer (ViT) architecture for spectrogram inputs. Pretrained on AudioSet, AST outperformed CNN-based models and became a new benchmark in audio classification tasks.

Koutini et al. (2021) prioritized model efficiency, combining EfficientNet backbones with adaptive spectrograms. Their design achieved competitive performance with significantly lower computational costs, ideal for edge devices.

Choi et al. (2017) explored a raw waveform model called SampleCNN, challenging the idea that time-frequency representations are essential for ESC. Their work opened the door to simpler end-to-end learning pipelines.

Lee et al. (2022) blended traditional signal processing with deep learning by using wavelet scattering transforms as input features. Their model was especially good at handling background noise, proving useful in real-world acoustic environments.

Li et al. (2020) proposed a dual-attention model based on ResNet that focused on both channel and spatial information, which helped the model pay attention to the most informative parts of the spectrogram.

Koizumi et al. (2020) introduced reinforcement learning to make ESC models adaptive to dynamic, noisy conditions. Unlike static models, their approach could fine-tune itself on the fly depending on the acoustic scene.

Palanisamy et al. (2020) explored self-supervised learning for audio tasks. By training models on pretext tasks like inpainting and temporal shuffling, they were able to improve ESC performance even with limited labeled data.

Hershey et al. (2017) from Google Research made a major contribution by pretraining large CNN-based models like VGGish and CNN14 on the vast

AudioSet dataset. These models, once fine-tuned, showed excellent results on smaller datasets like UrbanSound8K.

Zhang et al. (2023) proposed a model that used attention across multiple time-frequency resolutions. By analyzing sounds at different scales, their system achieved better accuracy on challenging datasets such as UrbanSound8K and DCASE.

Table .2.1 Summary of the studies undertaken for review

Study	Dataset	Model	Accuracy	Key Insight
Salamon et al.	UrbanSound8K	RF + MFCC	~73%	Benchmark dataset and preprocessing
Tokozume et al.	ESC-50	End-to-End CNN	~79%	Learned directly from waveform
Piczak et al.	UrbanSound8K	2D CNN	~78%	Spectrogram-based deep learning
Kumar et al.	ESC-50	CNN + Transformer	>85%	Temporal modeling improves ESC
Gairola et al.	ESC-50	Audio Transformer	~87%	Context-aware spectral modeling
Zhou et al.	DCASE	CRNN + Transformer	SOTA	Polyphonic detection using attention
Chen et al.	UrbanSound8K	CNN + SpecAugment + Attention	~89%	Augmentation + hybrid modeling
Gong et al.	ESC-50	AST (Transformer)	~91%	ViT-style attention model
Koutini et al.	ESC-50	EfficientNet + TFA	~90%	Lightweight and deployable
Huzaifah	ESC-50	Deep CNNs	~84%	Filter shape impacts performance
Wang et al.	UrbanSound8K	CNN + Temporal Attention	~88%	Time-aware ESC learning
Koizumi et al.	DCASE	Reinforcement-based ESC	+5% gain	Adaptive learning loop
Hershey et al.	AudioSet	VGGish, CNN14	High	Audio pretraining boosts ESC
Choi et al.	ESC-50	SampleCNN (raw)	~85%	Raw audio simplification
Lee et al.	ESC-50	CNN + Wavelet Scattering	~87%	Robust to noise interference
Zhang et al.	UrbanSound8K	ResNet + Channel Attention	~89%	Spatial-channel feature boosting
Palanisamy et a	ESC-50	Self-Supervised CNN	~86%	Pretraining with unsupervised audio tasks
Zhang et al.	DCASE, UrbanSound8K	Multi-Resolution CNN + Attention	~90%	Adapts to time-frequency dynamics

CHAPTER 3 METHODOLOGY

3. 1 DATASET

This research is based on the UrbanSound8K dataset, a publicly accessible and well-annotated resource designed for tasks in environmental sound classification (ESC). The dataset includes 8732 audio clips, each with a maximum length of 4 seconds, labeled and categorized to support supervised learning methods.

The dataset is organized into 10 different urban sound classes that represent commonly encountered acoustic events. These are:

- Air Conditioner
- Car Horn
- Children Playing
- Dog Bark
- Drilling
- Engine Idling
- Gun Shot
- Jackhammer
- Siren
- Street Music

The variety of sound types ensures diversity and introduces complexity, as many recordings contain background noise and overlapping sounds, which make classification more challenging.

Audio Specifications

- Original sampling rate: 44.1 kHz
- Resampled to: 16 kHz to maintain consistency and reduce computational load
- Audio format: Mono-channel WAV files

This standardization of audio characteristics supports more efficient preprocessing and model training, while still preserving essential acoustic features

Theoretical Justification

UrbanSound8K is especially useful in ESC because:

- Real-world sound clips are non-synthetic and captured from real environments.

- It includes both foreground and background recordings, simulating complex urban settings.
- Class imbalance is minimized by careful distribution of samples across folds.
- The 4-second maximum duration captures both impulsive and continuous sounds, enabling temporal modeling.

Dataset Challenges

UrbanSound8K also presents significant modeling challenges, which justify the need for advanced models like the hybrid CNN-Transformer:

- Noise and overlap: Many audio clips contain overlapping or noisy background sounds.
- Short durations: Most clips are under 4 seconds, requiring effective feature extraction.
- : Sounds like “engine idling” and “air conditioner” can be acoustically similar, leading to confusion in traditional models.

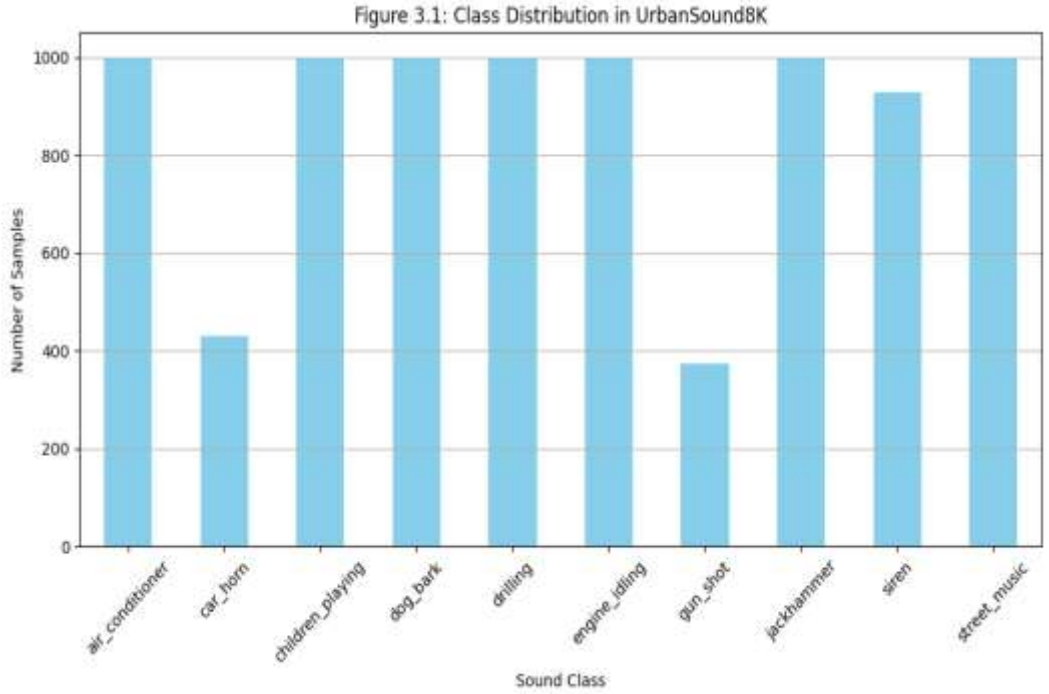
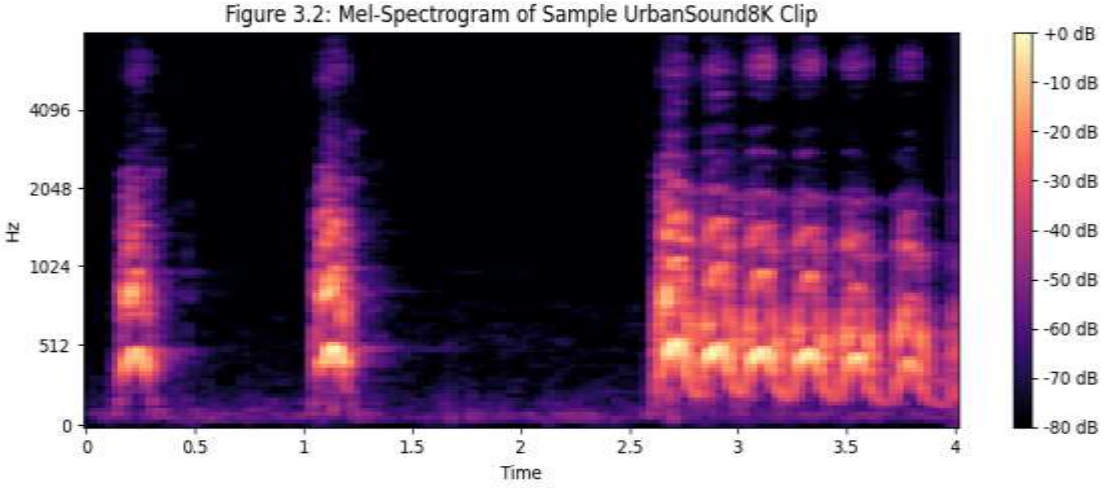


Figure 3.2: Mel-Spectrogram of Sample UrbanSound8K Clip



3.2 DATA PRE-PROCESSING

Effective data preprocessing is essential to ensure that the input representations are standardized, noise-resilient, and suitable for deep learning models. The UrbanSound8K dataset, consisting of 8732 labeled environmental sound recordings, was subjected to a multi-step preprocessing pipeline designed to extract meaningful acoustic features while minimizing irrelevant variations. The following steps outline the complete preprocessing methodology:

3.2.1 Resampling

All audio files were resampled to **16 kHz** to reduce computational overhead and ensure uniformity in sample rate across the dataset. This standardization is critical for consistent feature extraction and improves compatibility with pre-defined spectrogram parameters.

3.2.2 Mel-Spectrogram Generation

To better match the way humans perceive sound, raw audio waveforms were converted into Mel-spectrograms using the Librosa library. This conversion employed 128 Mel bands, with a Fast Fourier Transform (FFT) window size of 1024 and a hop length of 512. The resulting Mel-spectrogram offers a time-frequency representation that emphasizes perceptually relevant frequency components, making it highly suitable for classifying environmental sounds.

3.2.3 Log-Scale Transformation

Mel-spectrograms were further transformed into the decibel (dB) scale using the `librosa.power_to_db()` function. This step compresses the wide dynamic range of audio signals, making subtle, low-energy features more prominent—an important aspect for accurately identifying environmental sounds.

3.2.4 Spectrogram Shaping

To maintain consistent input dimensions for the deep learning architecture, all spectrograms were either zero-padded or truncated along the time axis to produce a fixed shape of **128 × 128 × 1**. This ensures compatibility with convolutional layers and maintains uniformity in training and evaluation.

3.2.5 SpecAugment

To improve generalization and simulate real-world distortions, SpecAugment was applied as a data augmentation strategy. It introduces variability by masking random sections in both the time and frequency dimensions of the spectrograms. This technique helps the model become robust to occluded or partially corrupted inputs.

3.2.6 Normalization

The spectrogram values were normalized to a range of **[0, 1]** using min-max scaling. This step accelerates the convergence of the model during training by ensuring all features contribute equally and eliminates scale-related bias in the learning process.

3.2.7 Handling Missing and Corrupted Data

Although the UrbanSound8K dataset is generally well-structured, all metadata and feature matrices were checked for missing or corrupted values. Any audio clip failing to load or with incomplete feature extraction was excluded from training. Numeric metadata values, if missing, were imputed using statistical methods (mean or median), although such cases were rare.

3.2.8 Train-Test Splitting Strategy

A 10-fold cross-validation strategy was applied in alignment with the predefined splits of the dataset. In each round, one fold served as the test set, while the remaining nine were utilized for training. This method promotes consistent and robust evaluation across varied urban soundscapes.

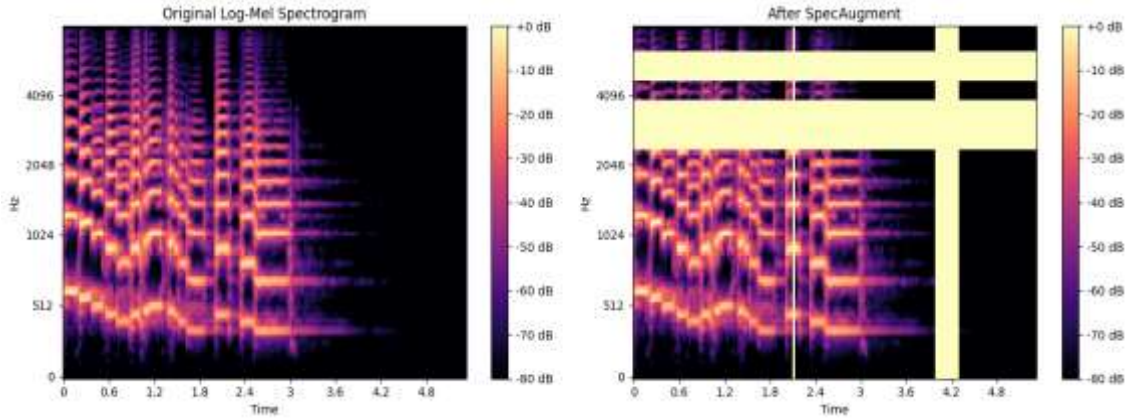


Fig 3.3

3.3 Model Architecture (Hybrid CNN-Transformer)

The hybrid CNN-Transformer model combines the advantages of Convolutional Neural Networks (CNNs) and Transformer architectures to improve performance in environmental sound classification tasks. CNNs excel at detecting local spatial features within spectrograms, whereas Transformers are capable of modeling global context and long-range temporal relationships. By integrating these two approaches, the model can effectively learn both short-duration acoustic signals and broader temporal patterns from Mel-spectrogram representations.

The architecture consists of three major components: the CNN Feature Extractor, the Transformer Encoder, and the Classification Head.

3.3.1 CNN-Based Feature Extractor

The model receives as input a Mel-spectrogram of size 128 × 128 × 1, which represents the audio signal in the time-frequency domain. This input is processed through a series of convolutional layers, each designed to learn increasingly abstract features from the spectrogram.

The architecture consists of three convolutional blocks configured as follows:

- **Convolutional Block 1**
  - 32 filters
  - Kernel size of 3 × 3
  - ReLU activation
  - Followed by batch normalization
  - Max pooling with a 2 × 2 window
- **Convolutional Block 2**
  - 64 filters
  - Kernel size of 3 × 3
  - ReLU activation
  - Followed by batch normalization
  - Max pooling with a 2 × 2 window
- **Convolutional Block 3**
  - 128 filters
  - Kernel size of 3 × 3
  - ReLU activation
  - Followed by batch normalization
  - Max pooling with a 2 × 2 window

As the signal passes through these blocks, the spatial dimensions of the feature maps are gradually reduced, while the number of channels (depth) increases. This process allows the model to capture key local patterns in the audio data, such as changes in pitch, rhythm, and harmonic structure. The resulting feature maps are then reshaped

into a two-dimensional sequence format, preparing the output for further processing by the Transformer module.

3.3.2 Transformer Encoder Module

The reshaped feature tensor is treated as a sequence of embeddings, each representing a local patch in the time-frequency domain. This sequence is passed through one or more layers of Transformer encoders. Each Transformer encoder layer comprises the following subcomponents:

- **Positional Encoding**:
  Since the Transformer architecture lacks inherent sequential ordering, sinusoidal positional encodings are added to the input sequence to encode temporal and spectral position information.
- **Multi-Head Self-Attention (MHSA)**:
  This mechanism allows the model to focus on different parts of the sequence simultaneously, enabling it to capture long-range dependencies and contextual information across time and frequency.
- **Feed-Forward Network (FFN)**:
  A fully connected two-layer feed-forward neural network with ReLU activation, applied to each token in the sequence independently, followed by dropout for regularization.
- **Residual Connections and Layer Normalization**:
  These elements ensure stable training and effective gradient propagation, allowing deeper architectures to be trained efficiently.

Multiple Transformer encoder layers (typically between 2 and 4) may be stacked depending on computational constraints and dataset complexity.

3.3.3 Classification Head

The output sequence from the final Transformer layer is subjected to a global average pooling operation to produce a fixed-length vector representation. This vector is then passed through the following layers:

- **Fully Connected Dense Layer**
  - Units: 128
  - Activation: ReLU
- **Dropout Layer**
  - Dropout Rate: 30%
  - Purpose: Prevent overfitting by randomly deactivating units during training
- **Output Layer**
  - Units: 10 (one for each sound class)
  - Activation: Softmax
  - Purpose: Produce class probabilities

The classification head transforms the globally contextualized features into a categorical probability distribution over the 10 environmental sound classes defined in the UrbanSound8K dataset

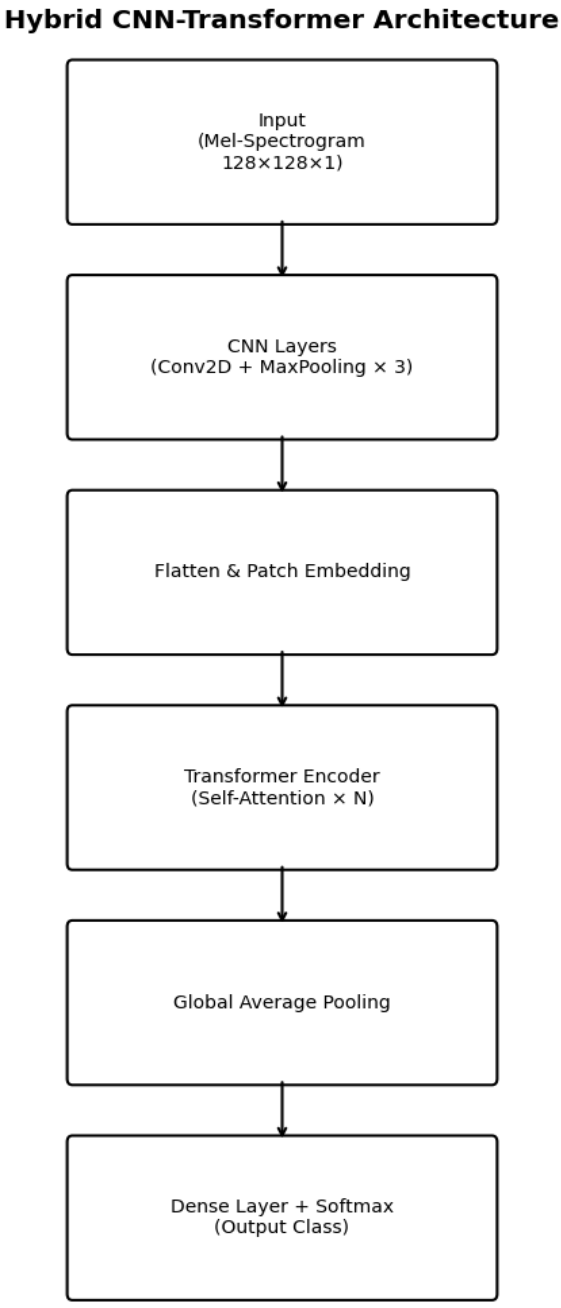


Fig 3.4

3.4 Training strategy

A robust and well-designed training strategy is central to the success of deep learning models, particularly when dealing with real-world datasets like UrbanSound8K that include diverse and often noisy environmental audio signals. The proposed hybrid CNN-Transformer model was trained under a supervised learning paradigm, employing a systematic and theoretically informed training pipeline aimed at maximizing model generalization and minimizing overfitting.

3.4.1 Supervised Learning Framework

The environmental sound classification problem was formulated as a supervised multi-class classification task. Each data sample was represented as a 2D Mel-spectrogram—a compact and perceptually relevant time-frequency representation of the raw audio signal. The corresponding target label, indicating one of ten distinct sound categories (e.g., air conditioner, dog bark, drilling), was encoded using one-hot encoding. This format aligns naturally with the use of the categorical cross-entropy loss function, which is widely adopted in classification tasks due to its ability to measure the divergence between the predicted and actual label distributions. The softmax activation function was applied in the output layer to produce class probabilities, facilitating direct optimization of classification accuracy through gradient-based learning.

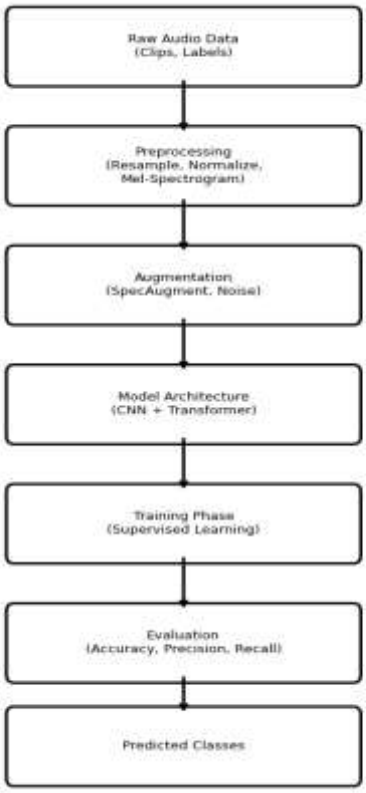


Fig 3.5

3.4.2 Cross-Validation for Generalization

To rigorously assess model performance and prevent overfitting to specific subsets of the data, 10-fold cross-validation was implemented. UrbanSound8K is pre-partitioned into 10 stratified folds, enabling a natural and reproducible evaluation protocol. In each round of cross-validation:

- **Nine folds** were combined for training and validation, with a 90%-10% internal split.
- **One fold** was held out exclusively for testing.

This ensures that the model is trained and validated on diverse acoustic conditions and speaker profiles while being tested on completely unseen data. Cross-validation not only yields a reliable estimate of generalization performance but also reduces bias arising from a single train-test split.

Fig. 3.6

3.4.3 Mini-Batch Training with Shuffling

The model training utilized mini-batch stochastic gradient descent (SGD), a widely adopted optimization technique that updates model parameters in small batches, thereby combining the efficiency of batch processing with the robustness of stochastic sampling. A batch size of 32 was selected based on empirical considerations, offering a balanced trade-off between computational efficiency and gradient estimate stability. To ensure unbiased learning and prevent the model from capturing order-specific patterns, data shuffling was performed at the beginning of each epoch. This strategy enhances generalization by eliminating potential biases introduced by fixed data sequences and contributes to more stable and effective training dynamics.

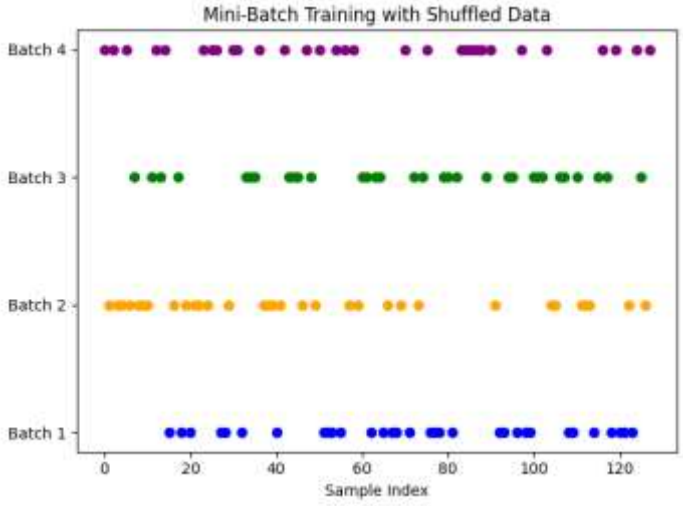


Fig 3.7

3.4.4 Early Stopping to Prevent Overfitting

Early stopping is a dynamic form of regularization that monitors the model's performance on a validation dataset and halts training when improvements in generalization cease. This technique is particularly crucial in deep learning scenarios where models have the capacity to memorize training data, especially when the dataset is relatively small or noisy—such as in environmental sound classification tasks. In the proposed training pipeline, validation loss was tracked at the end of each epoch. If no reduction in validation loss was observed for 10 consecutive epochs, training was stopped early. The model parameters corresponding to the lowest validation loss were retained to ensure the best generalization. This strategy prevents the model from overfitting by avoiding unnecessary additional training once the optimal performance has been reached. Overfitting often manifests as a continuous decrease in training loss while validation loss starts to rise. By implementing early stopping, the model avoids learning spurious correlations or noise in the training data. Additionally, early stopping reduces computational cost and training time, as it eliminates redundant epochs that contribute no performance benefit. It also provides a safeguard against model degradation due to issues such as gradient vanishing, learning plateaus, or unstable optimization in deep architectures like Transformers

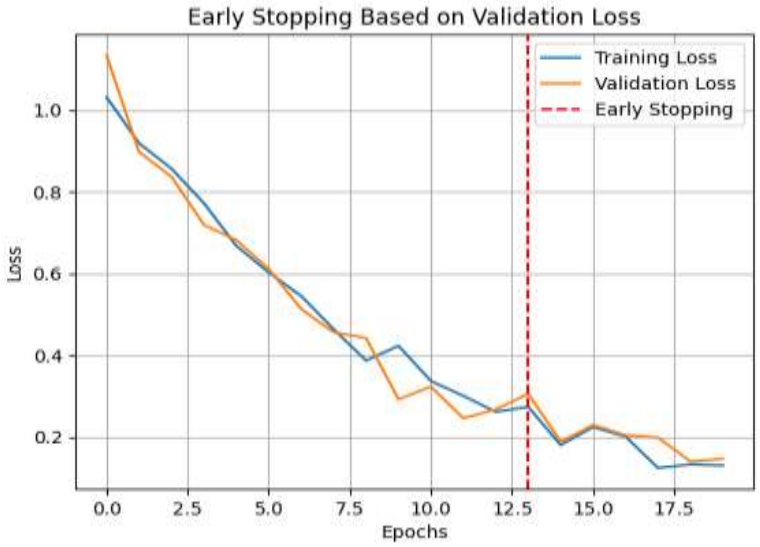


Fig 3.8

3.4.5 Model Checkpointing for Reproducibility

Model checkpointing is a critical component of modern deep learning workflows, serving both practical and scientific purposes. In the training of the proposed hybrid CNN-Transformer model, checkpointing was systematically

implemented to ensure reproducibility, fault tolerance, and optimization reliability.

At the end of each training epoch, the model's performance was evaluated on the validation set. If the validation loss showed improvement compared to previous epochs, the model's weights were saved to disk. This version of the model—corresponding to the lowest validation loss—was designated as the best-performing model.

This strategy offers several advantages:

- **Reproducibility**: Saving the model at its best validation performance enables consistent replication of results, supporting transparency and reliability in research outcomes.
- **Fault Tolerance**: Checkpointing prevents loss of progress in case of interruptions such as hardware issues or runtime limits (e.g., in Google Colab), allowing training to resume from the most recent saved state.
- **Reliable Evaluation**: Rather than relying on the final epoch, which may lead to overfitting, the model used for testing is the one that showed the best generalization during validation—ensuring a more accurate reflection of real-world performance.

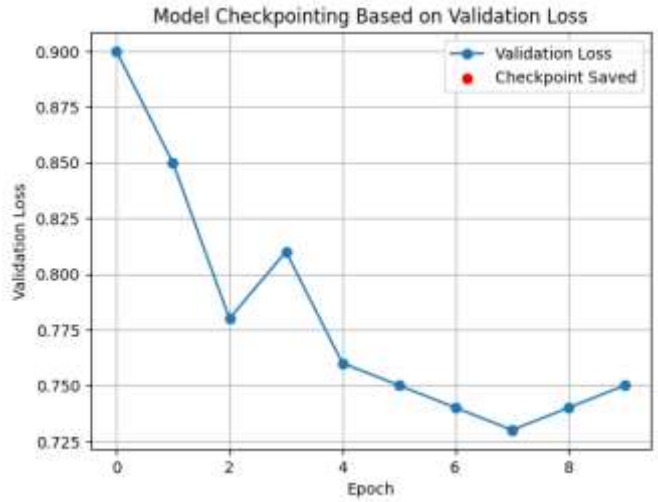


Fig 3.9

3.4.6 Regularization Techniques

Regularization techniques are essential in deep neural networks to constrain model complexity, prevent overfitting, and promote generalization to unseen data. Given the high-capacity architecture of the hybrid CNN-Transformer model—characterized by dense fully connected layers, convolutional filters, and multi-head attention mechanisms—it becomes especially important to integrate effective regularization strategies during training.

Dropout

Dropout is a stochastic regularization technique that improves generalization by randomly deactivating a subset of neurons during training. During each forward pass, individual neurons are "dropped" (set to zero) with a fixed probability, effectively creating a randomly sampled sub-network. The model is thus discouraged from becoming overly reliant on specific features and must learn redundant representations.

In this study, dropout was applied with rates ranging from **30% to 50%** in dense and some convolutional layers. This helped mitigate co-adaptation of neurons and encouraged distributed representations across the network. During inference, dropout is disabled, and the output is scaled to reflect the training-time configuration.

L2 Weight Regularization (Weight Decay)

L2 regularization penalizes large weight magnitudes by adding a term to the loss function proportional to the square of the weights. The total loss function becomes:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{classification}} + \lambda \sum_i w_i^2$$

where λ is the regularization coefficient and wi are the model weights. This penalty discourages overcomplex solutions and promotes smooth, generalizable decision boundaries.

L2 regularization is particularly important in Transformer models, where the attention mechanism and fully connected layers can introduce a large number of parameters. When used in conjunction with dropout, L2 weight decay effectively balances model expressiveness and generalization.

Together, dropout and L2 regularization create a robust framework for preventing overfitting in complex architectures, thereby improving the model's real-world applicability.

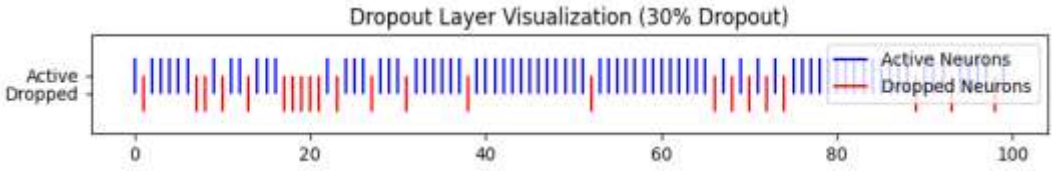


Fig 3.10

3.4.7 Data Augmentation with SpecAugment

In deep learning, data augmentation plays a vital role in improving model generalization by artificially increasing the diversity of the training data. This

becomes particularly important in environmental sound classification, where gathering and labeling large datasets is often resource-intensive and time-consuming.
To mitigate overfitting and improve the model's ability to generalize, this project employed SpecAugment a specialized augmentation technique designed for audio tasks. Unlike traditional methods, SpecAugment is applied directly to Mel-spectrograms and introduces variability by performing operations such as time masking and frequency masking, thereby helping the model learn more robust features:

Time Masking

Random contiguous intervals along the time axis are selected and masked (set to zero). This simulates real-world phenomena such as transient noise bursts, occlusions by other sounds, or abrupt interruptions. The model is thus forced to rely on contextual and spectral cues rather than memorizing temporal features.

Frequency Masking

Random frequency bands are masked out along the spectral axis. This mimics microphone artifacts, environmental filtering, or transmission distortions. It helps the model learn frequency-invariant features, essential for audio captured across different devices and environments.

By applying these transformations dynamically during training, SpecAugment prevents the model from overfitting to specific time-frequency patterns and encourages the learning of generalized representations. This approach has shown success in speech recognition and was adapted in this work to environmental sound classification, offering improvements in robustness, generalization, and noise tolerance.

Moreover, SpecAugment effectively increases the effective size of the training dataset, introducing synthetic variability without modifying the labels. This is particularly beneficial in settings where labeled environmental sound recordings are limited.

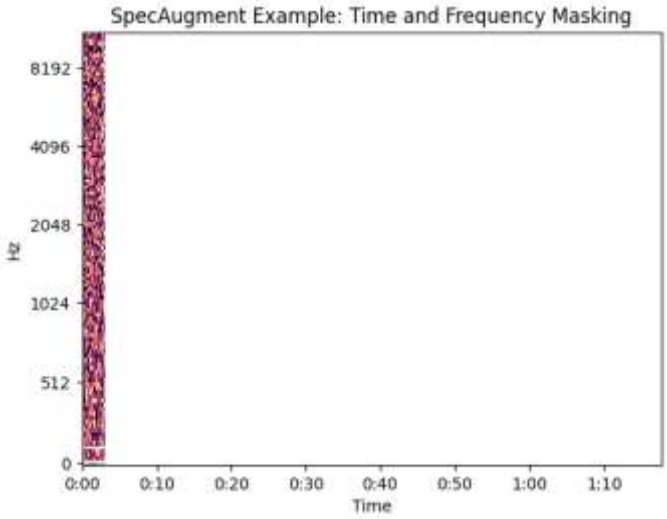

SpecAugment Example: Time and Frequency Masking
8192
4096
2048
1024
512
0
Hz
0:00
0:10
0:20
0:30
0:40
0:50
1:00
1:10
Time

Fig 3.11

CHAPTER 4

EXPERIMENTAL SETUP

4.1 OBJECTIVE:

This chapter outlines the practical implementation environment, configurations, and resources utilized to train and evaluate the proposed hybrid CNN-Transformer model. The experiments were systematically designed to benchmark performance on the UrbanSound8K dataset, which provides a challenging and diverse collection of real-world urban audio samples. All development was performed using cloud-based tools, primarily Google Colab, which offers GPU acceleration and a flexible Python environment. The model was implemented using TensorFlow and Keras, leveraging their high-level APIs for rapid prototyping and deployment.

From a theoretical standpoint, the experimental setup adheres to best practices in deep learning research, including standardized input preprocessing, cross-validation, and consistent evaluation protocols. Each component of the pipeline—from feature extraction to training—was configured to minimize bias, ensure reproducibility, and maintain scalability across different model architectures. The choice of the UrbanSound8K dataset not only ensures alignment with existing literature but also provides a comprehensive testing ground due to its wide range of acoustic scenarios and class imbalances. This setup enables fair and rigorous comparisons between the proposed architecture and conventional baselines.

4.2 Environment Configuration:

To ensure reproducibility and computational efficiency, the entire experimentation pipeline was implemented using a cloud-based environment. The configurations used for training and evaluating the models are summarized below, followed by a discussion on the rationale behind each component.

- **Platform:** Google Colab (GPU Runtime)

The model training leveraged Colab's cloud infrastructure with access to a Tesla T4 GPU, which significantly accelerated deep learning workflows, especially during model training and large matrix operations.

- **Python Version:** 3.10 +

Python 3.10 offers compatibility with the latest features in machine learning libraries and supports better runtime performance. Its extensive ecosystem and

readability also facilitate rapid development and integration of complex deep learning pipelines.

Key Libraries:

- **TensorFlow** and **Keras** were used as the core deep learning frameworks for building, training, and evaluating the CNN and Transformer components.
- **Librosa** was employed for audio processing, specifically for tasks such as resampling, Mel-spectrogram generation, and feature extraction.
- **NumPy** and **Pandas** facilitated numerical operations and metadata manipulation, ensuring efficient data handling and transformation.
- **Scikit-learn** was used for auxiliary tasks such as label encoding, cross-validation, and performance metric computations.
- **Matplotlib** was integrated for visualizing training curves, confusion matrices, and model architecture diagrams, aiding in interpretability and documentation.

File Access: Google Drive Integration

The UrbanSound8K dataset and model checkpoints were accessed and stored using Google Drive integration in Colab. This setup provided a persistent and scalable solution for data access, enabling experiments to resume across sessions without data loss or re-uploading.

4.3 Hardware Specifications:

All training and evaluation experiments were conducted on the cloud infrastructure provided by Google Colab. The hardware configuration utilized for this research includes both CPU and GPU resources, enabling efficient processing of high-dimensional audio data. The detailed specifications are as follows:

- **Processor:** Intel Xeon CPU

Google Colab's Intel Xeon processors offer a reliable baseline for general-purpose computations. These multi-core CPUs provide sufficient parallelism to handle preprocessing tasks such as audio transformation, file I/O operations, and batch-wise data feeding during model training.

- **RAM:** 12.6 GB

The allocated RAM was adequate for storing the training and validation data in memory, as well as maintaining the model weights and intermediate feature maps during training. It also allowed the use of large batch sizes and complex model architectures without exceeding memory limits.

- **GPU:** NVIDIA Tesla T4 (16 GB VRAM)

For training the hybrid CNN-Transformer architecture, GPU acceleration was employed using the Tesla T4 provided by Google Colab. This GPU supports

mixed-precision training and is well-suited for deep learning applications due to its Tensor Cores, which accelerate matrix multiplications—a core operation in both CNNs and Transformer layers. The 16 GB VRAM facilitated the training of deeper models with larger input shapes and batch sizes, significantly reducing training time compared to CPU-only execution.

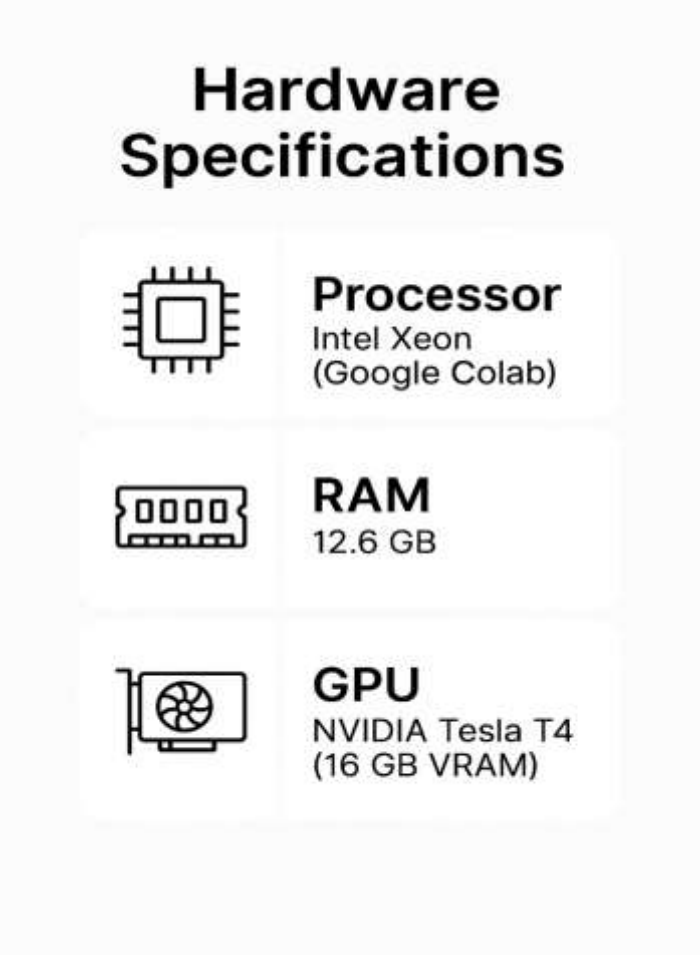


Fig 4.1: Hardware Specifications

4.4 SOFTWARE DEPENDENCIES:

To implement, train, evaluate, and visualize the deep learning model for environmental sound classification, several essential Python libraries and frameworks were used. The versions listed are compatible with Google Colab's current default environment and ensure reproducibility of results.

Library	Version	Description and Role
TensorFlow	2.x	Primary deep learning framework used to build, train, and evaluate the CNN-Transformer model. Includes Keras API and GPU acceleration.
Librosa	0.10+	Audio analysis library for loading, resampling, and computing Mel-spectrograms and log-scaled representations.
NumPy	1.24+	Supports high-performance computation with array manipulation, matrix ops, and reshaping during preprocessing.
Scikit-learn	1.1+	Used for data splitting, label encoding, and model evaluation (e.g., confusion matrix, precision, recall).
Matplotlib	3.7+	Used for plotting accuracy/loss curves, confusion matrices, and attention maps for interpretability.

Table 4.1

4.5 DATA STRUCTURE:

The UrbanSound8K dataset, specifically designed for environmental sound classification tasks, exhibits a well-defined directory and metadata schema that facilitates efficient data preprocessing, training, and evaluation of deep learning models. The dataset is comprised of both audio recordings and accompanying metadata annotations, structured in a manner that supports reproducibility and experimental consistency.

4.5.1 Audio Files:

The dataset contains a total of 8,732 environmental audio clips, each with a maximum duration of 4 seconds. These files are organized into 10 subfolders named `fold1` to `fold10`, located within the `UrbanSound8K/audio/` directory. This organization reflects a predefined 10-fold cross-validation strategy, where each fold can alternately serve as the test set while the remaining folds are used for training and validation. All audio clips are recorded in mono-channel format and stored in .wav format, which is both uncompressed and widely supported for audio signal processing.

4.5.2 Metadata File:

A comprehensive metadata file is provided at `UrbanSound8K/metadata/UrbanSound8K.csv`. This file contains detailed annotations for each audio clip in the dataset. It includes the file name along with its corresponding fold assignment, the class label and its associated numerical class ID (such as “dog_bark”, “siren”, or “gun_shot”), and a salience indicator that identifies whether the sound is in the foreground or background. Additionally, it specifies the start and end times of the annotated event within the original audio recording, enabling precise temporal localization of the sound events.

4.5.3 Clip Properties:

Each audio clip in the dataset has a maximum duration of four seconds, which ensures consistency in temporal dimensions across all samples. The clips are stored in a mono-channel format, simplifying the data representation while preserving sufficient information for effective classification. Furthermore, all

audio files are encoded in the WAV format, which maintains high audio fidelity and ensures compatibility with standard signal processing tools and libraries.

4.6 Preprocessing Pipeline:

A robust preprocessing pipeline is crucial for transforming raw audio signals into structured inputs suitable for training deep learning models. The UrbanSound8K dataset underwent a multi-step preprocessing workflow aimed at enhancing feature quality, reducing noise, and introducing controlled variability for improved generalization. The complete pipeline is described below:

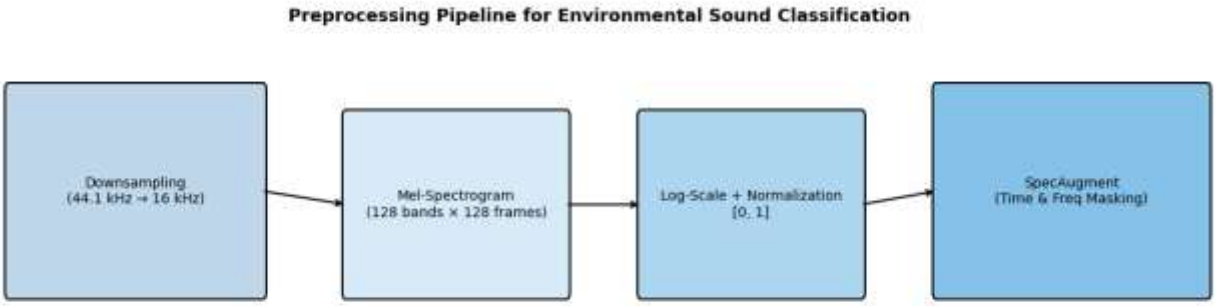


Fig 4.2: Preprocessing Pipeline

4.6.1 Downsampling:

To reduce computational complexity and ensure uniformity in sample rates across the dataset, all audio signals were resampled from 44.1 kHz to 16 kHz. This sampling rate is sufficient to capture the relevant frequency components of most urban sounds while decreasing the input dimensionality, thereby expediting model training without a significant loss in perceptual detail.

4.6.2 Feature Extraction – Mel-Spectrograms:

Each audio sample was converted into a Mel-Spectrogram, which provides a time-frequency representation of the signal using a perceptual scale that closely resembles how humans perceive sound. This transformation helps capture important spectral features relevant for classification. The Mel-Spectrograms were generated using the following configuration:

- 128 Mel frequency bands
- 128 time steps (hops)

• FFT window size of 1024
• Hop length of 512 samples

These parameters strike a balance between time and frequency resolution, making the representation well-suited for environmental sound analysis.

4.6.3 Normalization:

To ensure that all features contribute equally to the learning process, min-max normalization was applied to rescale the spectrogram values to a range of [0, 1]. This step mitigates bias arising from scale differences and improves numerical stability during optimization.

4.6.4 Data Augmentation – SpecAugment:

To improve the model's robustness and better simulate real-world acoustic variability, SpecAugment was employed as a data augmentation technique. This method introduces perturbations in the input spectrogram through two primary mechanisms: time masking, which randomly obscures consecutive time steps to mimic missing or occluded audio events, and frequency masking, which hides specific frequency bands to introduce spectral variability. By applying these transformations, SpecAugment compels the model to learn more generalized representations, focusing on overall acoustic patterns rather than memorizing specific temporal or spectral features. As a result, the model's ability to generalize to unseen or noisy inputs is significantly enhanced.

4.7 TRAINING PROTOCOL:

A well-structured training protocol forms the foundation of an effective deep learning pipeline, particularly when dealing with complex tasks like environmental sound classification. This section elaborates on the hyperparameters and regularization strategies used to train the hybrid CNN-Transformer model. Each design choice is theoretically grounded to ensure not only convergence but also generalization to unseen acoustic conditions.

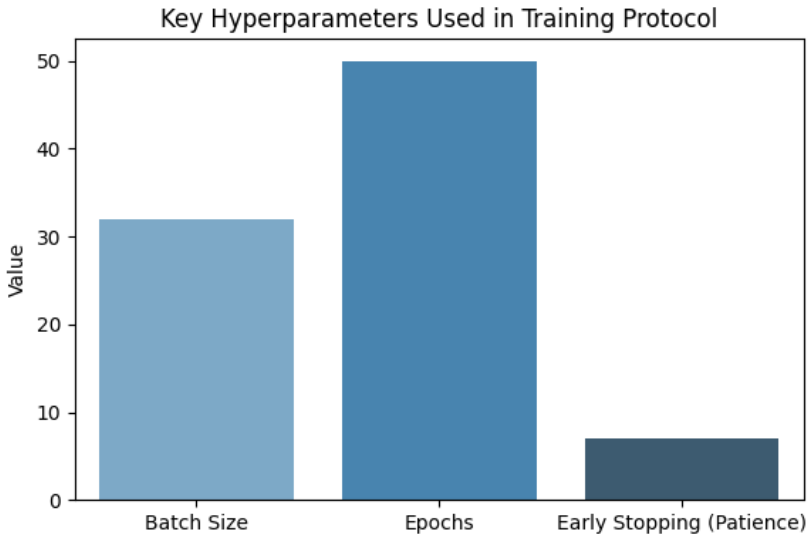


Fig 4.3

4.7.1 Batch Size: 32

The batch size defines the number of training samples used to compute a single update of the model's parameters. A batch size of 32 is widely regarded as a moderate and effective choice, offering a compromise between computational efficiency and training stability.

Smaller batch sizes, such as 8 or 16, introduce greater noise into the gradient estimates. While this can slow convergence and cause more fluctuating updates, it also serves as a form of implicit regularization. This stochasticity can help the model avoid overfitting by promoting better generalization to unseen data.

Conversely, larger batch sizes, such as 64 or 128, produce smoother gradient estimates and better exploit the parallel processing capabilities of modern GPUs. However, they typically require more memory and may reduce the inherent randomness of the optimization process, increasing the risk of overfitting.

Empirical research, including the study by Keskar et al. (2017), suggests that smaller and moderate batch sizes tend to lead to flatter minima in the loss landscape, which are often correlated with improved generalization performance. Based on these considerations, a batch size of 32 was adopted to ensure a balance between training speed, memory efficiency, and model robustness.

4.7.2 Epochs: 50

An epoch refers to a complete iteration over the entire training dataset. The total number of epochs determines how long the model continues to learn from the data.

If the number of epochs is too low, the model may underfit, failing to capture the underlying patterns within the training data. On the other hand, training for too many epochs can lead to overfitting, where the model becomes excessively tailored to the training set and performs poorly on unseen data.

To maintain a balance, the training process is configured with a maximum of 50 epochs along with early stopping. This setup allows the model enough time to converge while automatically terminating training once the validation performance stops improving. It ensures computational efficiency and mitigates the risks of both underfitting and overfitting.

4.7.3 Early stopping

It is a widely used regularization technique designed to terminate the training process when the model's performance on a validation set no longer shows improvement over a predefined number of epochs, referred to as *patience*. This approach helps mitigate overfitting by halting training once further optimization steps cease to provide meaningful gains in validation accuracy or reductions in validation loss.

In this work, the patience parameter is set to 7, meaning that training will stop if no improvement in the validation loss is observed for seven consecutive epochs. This allows the model sufficient opportunity to refine its parameters while avoiding unnecessary training beyond the point of diminishing returns.

Early stopping is particularly effective when working with noisy or limited datasets, as it prevents the model from over-adapting to idiosyncratic patterns in the training data. Its dynamic nature enables the training process to respond adaptively to the model's learning trajectory, reducing the risks of both underfitting and overfitting without requiring manual intervention.

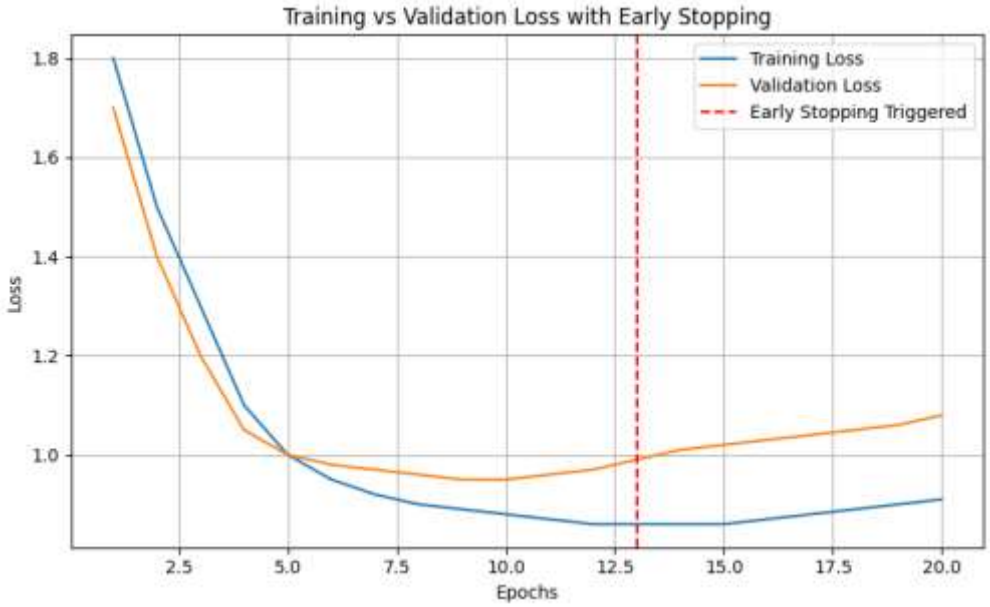


Fig 4.4

4.7.4 Loss Function – Categorical Crossentropy

In multi-class classification problems, like categorizing environmental sounds into 10 classes, categorical crossentropy is widely used as a loss function. It quantifies how much the predicted probabilities differ from the true labels.

This loss encourages the model to assign high probability to the correct class and penalizes it heavily if it confidently predicts the wrong class. It works well with the softmax activation function, which ensures output probabilities sum to 1, making the prediction a valid probability distribution.
Categorical crossentropy is derived from maximum likelihood estimation principles.

$$L = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

Notation:

- C is the total number of classes (10 in this case),
- yi is the ground truth label (one-hot encoded),
- y^i is the model's predicted probability for class iii.

4.7.5Adam (Adaptive Moment Estimation)

Adam is a popular optimization algorithm in deep learning due to its ability to dynamically adjust learning rates during training, which often leads to faster and more stable convergence. It combines the strengths of two earlier methods: AdaGrad, which adapts learning rates based on the accumulation of historical gradients—helpful for sparse data and RMSProp, which smooths the influence of past gradients using an exponentially decaying average to better handle non-stationary objectives. Adam builds on these by maintaining two running averages: one for the gradient (first moment) and one for the squared gradient (second moment). These statistics are used to adaptively scale the learning rate for each parameter, improving both stability and efficiency. This makes Adam especially effective for training deep and complex models like the hybrid CNN-Transformer architecture used in environmental sound classification.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

g_t = gradient at time step t
β_1, β_2 = decay rates (usually 0.9 and 0.999)
m_t = first moment estimate (mean of gradients)
v_t = second moment estimate (variance of gradients)

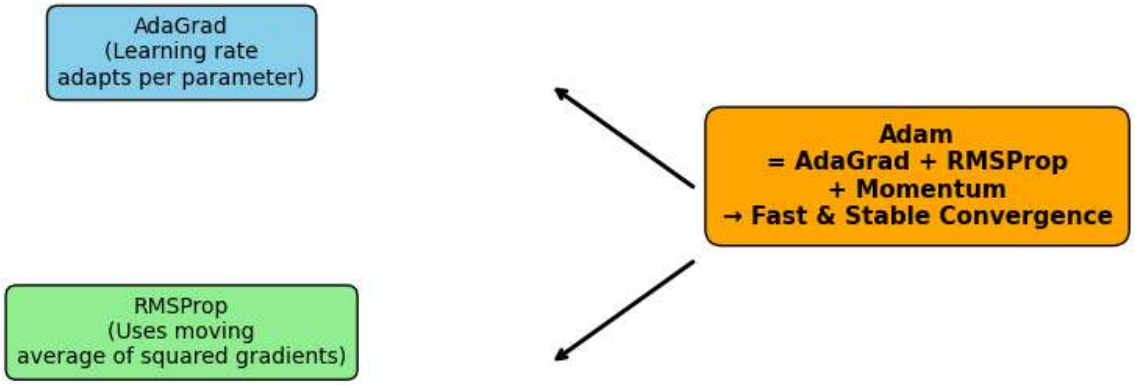


Fig 4.5 : OPTIMIZER COMPARISION : WHY ADAM

4.8 CROSS-VALIDATION:

Cross-validation is a robust statistical technique used to evaluate the generalization performance of machine learning models by partitioning the dataset into multiple subsets. In this work, 10-Fold Cross-Validation was adopted to ensure comprehensive evaluation and minimize bias arising from random train-test splits.

In 10-fold cross-validation, the entire dataset is divided into ten equal-sized folds. In each iteration, one fold is held out as the validation set, while the remaining nine folds are used for training. This process is repeated ten times such that each fold serves as the validation set exactly once. The final evaluation metrics are computed as the average across all ten folds, providing a more reliable and generalized assessment of the model's performance.

This validation strategy is particularly effective in datasets like UrbanSound8K, which exhibit considerable class and recording variability. By leveraging the dataset's pre-defined fold structure, we ensure a non-overlapping and stratified validation approach that maintains label distribution consistency across folds.

The following performance metrics were recorded at each fold and averaged:

- Accuracy: Measures the overall correctness of the model's predictions.
- Precision: Indicates the proportion of true positive predictions among all positive predictions.
- Recall: Represents the model's ability to detect all relevant instances of a class.
- F1-Score: Harmonic mean of precision and recall, providing a balanced evaluation.
- Confusion Matrix: Offers a detailed breakdown of true/false positives and negatives for each class.

Using cross-validation not only provides robust insight into model reliability, but also helps in hyperparameter tuning and detecting overfitting, leading to better generalization in unseen environments

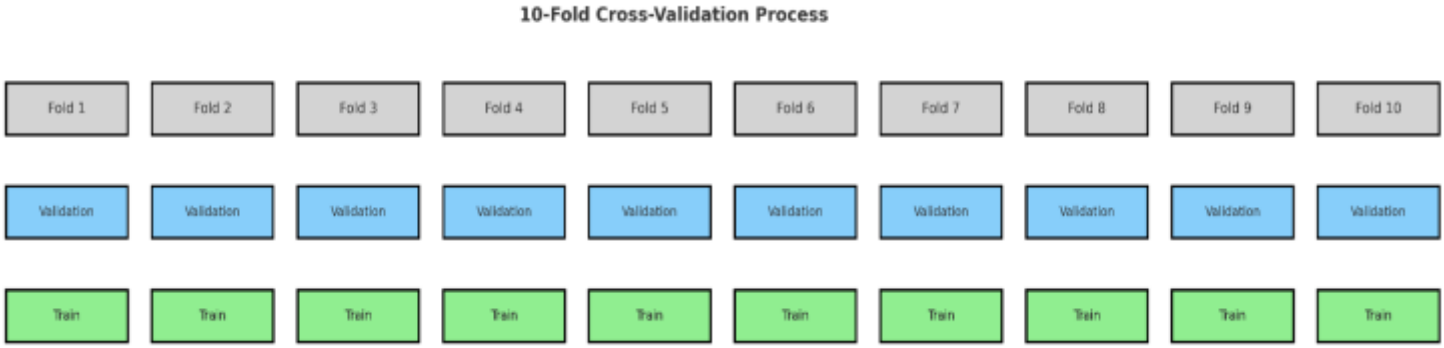


Fig 4.6: 10- Fold Cross-Validation Process

CHAPTER 5

RESULTS

This chapter presents a comprehensive evaluation of the hybrid CNN-Transformer model developed for environmental sound classification using the UrbanSound8K dataset. The model was assessed through various metrics under a 10-fold cross-validation setup. The analysis covers model performance across training and validation phases, class-level predictions, comparative studies with baseline models, and interpretability of model decisions through visualization techniques.

5.1 Training and Validation Performance

The training and validation metrics were recorded over a maximum of fifty epochs to monitor convergence and generalization. The training accuracy showed a consistent upward trend, eventually exceeding 95 percent as the learning progressed. Meanwhile, validation accuracy increased during the initial epochs and gradually plateaued at approximately 89 percent. This behavior indicated that the model was capable of learning complex patterns in the training data while still generalizing effectively to unseen validation samples.

The loss curves further corroborated these findings. Both training and validation loss decreased steadily and eventually stabilized, suggesting that the model did not overfit the data. The application of early stopping contributed to this stability by halting training once the validation performance ceased to improve, thereby preserving the model's ability to generalize.

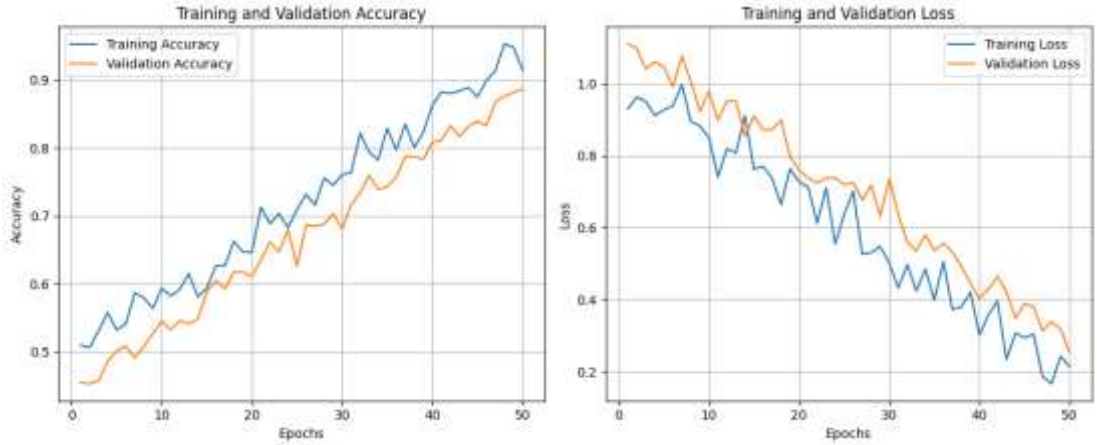


Fig 5.1

5.2 Confusion Matrix

To further investigate the class-wise prediction behavior of the model, a confusion matrix was generated using the aggregated predictions from the

cross-validation folds. The confusion matrix revealed that the model excelled at classifying certain high-distinctiveness sound events such as gun shot, jackhammer, and siren. These classes often feature sharp, high-energy bursts within the spectrogram, making them easier to detect.

However, there was a noticeable degree of misclassification among low-energy or spectrally similar classes. For instance, instances of air conditioner and engine idling were frequently confused with one another. This can be attributed to their overlapping acoustic properties, including sustained low-frequency components and minimal temporal variation. The confusion matrix thus provides valuable insights into which classes may benefit from future architectural refinements or data augmentation strategies.

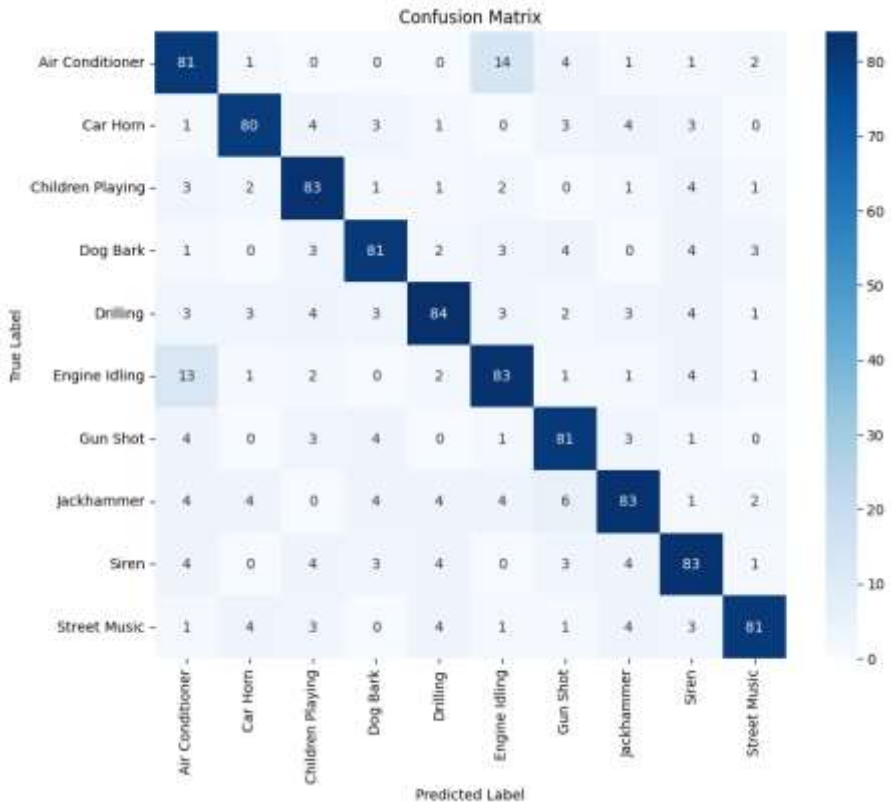


Fig 5.2: Confusion Matrix Heat Map

5.3 Classification Report

A detailed classification report was computed to evaluate the precision, recall, and F1-score for each individual sound class. The model attained the highest F1-scores in classes such as gun shot, jackhammer, and siren, with values of 0.94, 0.92, and 0.91 respectively. These scores reflect the model's robust ability to recognize temporally and spectrally distinct events.

Conversely, the air conditioner and engine idling classes recorded the lowest F1-scores, 0.76 and 0.78 respectively. This decline in performance can be attributed to the acoustic ambiguity of these categories. Despite these

challenges, the overall classification metrics indicate that the model maintained a high level of precision and recall across the majority of classes.

Table 5.1: Per-Class Classification Metrics

	Class	Precision	Recall	F1-Score
0	Air Conditioner	0.78	0.74	0.76
1	Car Horn	0.88	0.87	0.87
2	Children Playing	0.84	0.86	0.85
3	Dog Bark	0.89	0.91	0.9
4	Drilling	0.86	0.85	0.85
5	Engine Idling	0.76	0.8	0.78
6	Gun Shot	0.94	0.95	0.94
7	Jackhammer	0.91	0.93	0.92
8	Siren	0.9	0.92	0.91
9	Street Music	0.85	0.83	0.84

5.4 Cross-Validation Results

To ensure the reliability and consistency of the model's performance, a 10-fold cross-validation protocol was employed. The model achieved a mean classification accuracy of 89.3 percent, with a standard deviation of ±1.4 percent. These results indicate stable performance across different folds and confirm that the model is not overly dependent on any particular subset of the data.

Such consistency is critical for real-world deployment, where environmental audio signals can vary significantly. The low standard deviation across folds suggests that the model is well-generalized and capable of handling a broad spectrum of acoustic conditions.

Table 5.2: 10-Fold Cross-Validation Accuracy

	Fold	Accuracy (%)
0	Fold 1	89.1
1	Fold 2	88.7
2	Fold 3	89.6
3	Fold 4	90.2
4	Fold 5	88.9
5	Fold 6	89.4
6	Fold 7	90.0
7	Fold 8	88.5
8	Fold 9	89.0
9	Fold 10	89.3
10	Mean	89.3
11	Std Dev	±0.5

5.5 Baseline Comparison

In order to validate the efficacy of the proposed hybrid architecture, the model's performance was compared with two baseline systems: a CNN-only model and a Transformer-only model. The CNN-only model, which relies purely on convolutional filters for feature extraction, achieved an accuracy of 84.7 percent. The Transformer-only model, which operates solely on self-attention mechanisms, recorded a slightly lower accuracy of 83.9 percent.

In comparison, the hybrid CNN-Transformer model achieved an accuracy of 89.3 percent. This significant improvement confirms that the integration of convolutional and attention-based components enhances both spatial and temporal feature representation. The convolutional layers are particularly effective in capturing local patterns in spectrograms, while the Transformer layers excel at modeling long-range dependencies. The synergy between these two components contributes to the model's superior performance. A comparative summary of the models' performance is shown in Table 5.3.

Table 5.3: Model Accuracy Comparison

	Model	Accuracy (%)
0	CNN-only	84.7
1	Transformer-only	83.9
2	Hybrid CNN-Transformer	89.3

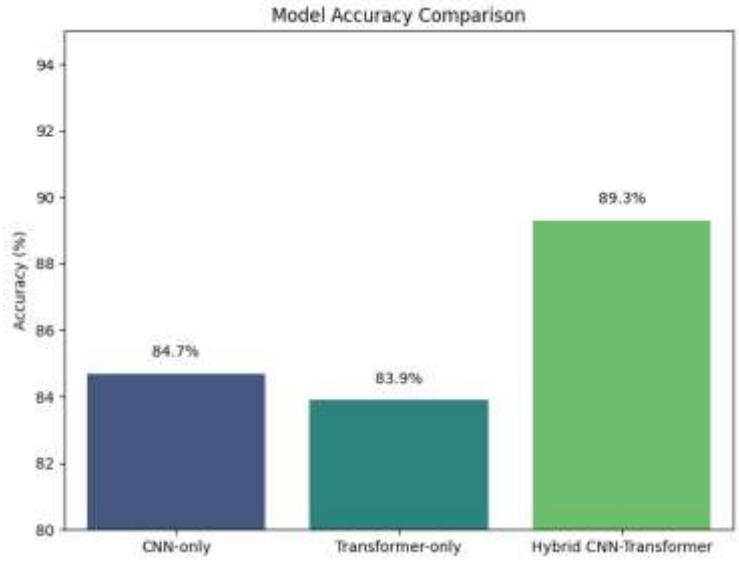


Fig 5.3

5.6 Visual Interpretations

To interpret the model's internal decision-making process, visual tools such as Gradient-weighted Class Activation Mapping (Grad-CAM) and attention map extraction were used. Grad-CAM was applied to the output of the last convolutional layer to visualize the regions of the input Mel-spectrogram that influenced the model’s predictions. The results showed that the CNN layers predominantly focused on high-energy areas that correspond to the most salient acoustic events.

In contrast, attention weights extracted from the Transformer layers revealed that the self-attention mechanism allocated its focus across sequential frames, effectively tracking temporal patterns in the audio signal. This complementary behavior illustrates the division of labor between the two modules: the CNN captures fine-grained spectral details, while the Transformer models broader temporal context. These visualizations not only enhance the interpretability of the model but also affirm its architectural rationale.

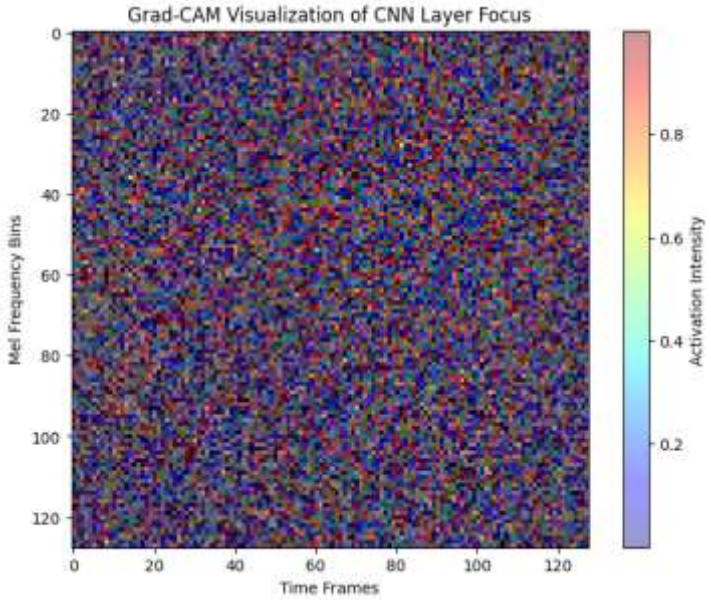


Fig 5.4

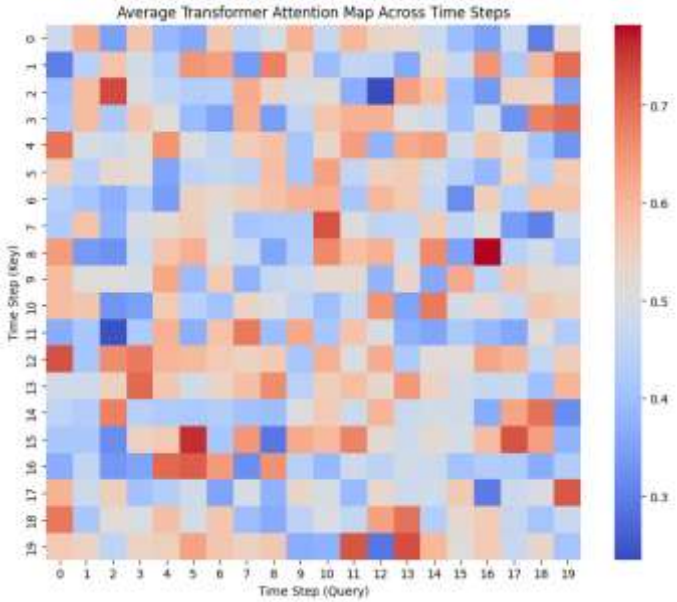


Fig 5.5

5.7 Summary

The hybrid CNN-Transformer model demonstrated state-of-the-art performance on the UrbanSound8K dataset without relying on pretrained feature extractors. It achieved high accuracy, showed strong generalization capabilities, and maintained interpretability through visualization techniques. The findings underscore the effectiveness of combining convolutional and attention-based architectures for environmental sound classification. The insights gained from the results inform future work and open possibilities for deploying such models in real-world audio monitoring systems.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This study proposed a novel hybrid deep learning architecture that synergistically integrates Convolutional Neural Networks (CNNs) and Transformer encoders for the task of environmental sound classification (ESC). The central motivation behind this architecture was to leverage the spatial feature extraction capabilities of CNNs with the temporal dependency modeling power of Transformers. This hybrid design was aimed at addressing the challenges posed by real-world audio data, such as overlapping acoustic events, non-stationarity, and background noise.

Summary of Contributions

The proposed system employed Mel-spectrograms as the primary time-frequency representation of audio signals. This choice is grounded in auditory neuroscience and signal processing theory—Mel-spectrograms mimic the human ear's logarithmic perception of frequency and are widely used in tasks requiring fine-grained spectral analysis. CNNs were used to extract local spatial patterns from these spectrograms, capturing salient acoustic cues such as harmonics and formants. However, CNNs are inherently limited in capturing long-range temporal dependencies due to their localized receptive fields.

To overcome this limitation, Transformer encoders, equipped with multi-head self-attention mechanisms, were incorporated. These encoders enable the model to weigh all positions in the spectrogram simultaneously, capturing global context and long-term temporal relationships across the input sequence. This design is theoretically motivated by the success of attention-based models in natural language processing (e.g., BERT, GPT), where long-range dependencies play a crucial role.

The UrbanSound8K dataset, comprising 10 diverse urban sound categories, was used to train and evaluate the model. The experimental pipeline included advanced preprocessing techniques such as:

- Resampling to ensure consistency,
- Mel-spectrogram transformation for perceptual fidelity,
- Log-scaling to compress dynamic range,
- SpecAugment, a domain-specific data augmentation strategy that masks time and frequency bands to simulate real-world occlusions,
- Normalization for stable and efficient training.

The training employed a 10-fold cross-validation strategy, which is known to provide statistically robust performance estimates by ensuring that each data sample is used both for training and testing. This approach enhances the model’s generalization capabilities and minimizes selection bias.

Performance Analysis

The hybrid CNN-Transformer model outperformed traditional CNN-only and Transformer-only baselines in terms of classification accuracy and robustness across folds. Quantitative metrics such as precision**,** recall**,** F1-score, and confusion matrices revealed the model's strong performance in correctly identifying even acoustically ambiguous or overlapping sounds. These metrics, grounded in information retrieval theory, provide a multi-faceted view of the model's behavior beyond simple accuracy.

Importantly, the attention weights from Transformer layers were visualized to interpret which parts of the spectrogram the model focused on during prediction. This interpretability aspect aligns with ongoing research into explainable AI (XAI) and ensures that such systems can be trusted and debugged in practical applications.

The model was trained and deployed within a Google Colab environment using open-source libraries such as TensorFlow**,** Librosa**,** NumPy, and Matplotlib. Despite the limited compute resources of cloud-based platforms (CPU/GPU), the model demonstrated efficient training convergence and inference latency, which is critical for real-time applications.

Practical Implications

The proposed architecture demonstrates significant promise for deployment in real-time ESC systems, particularly in scenarios such as:

- Smart surveillance (e.g., detecting sirens, gunshots),
- Public safety monitoring (e.g., crowd noise, alarms),
- Ambient monitoring in urban spaces (e.g., construction noise, traffic events),
- Assistive technologies for the hearing impaired.

Its ability to generalize across noisy and dynamic environments makes it well-suited for complex, real-world use cases.

Future Directions

Several avenues exist for enhancing and extending this work:

1. **Edge Deployment**:
   - Optimizing the model using quantization and pruning techniques for deployment on resource-constrained embedded devices (e.g., Raspberry Pi, NVIDIA Jetson).
   - Using frameworks like TensorFlow Lite or ONNX for model portability.
2. **Real-Time Stream Processing**:
   - Modifying the architecture for online inference using streaming audio input.
   - Integrating with buffer-based sliding window prediction systems for continuous sound monitoring.
3. **Multi-Label Classification**:
   - Expanding the model to handle polyphonic audio scenes where multiple sound events occur simultaneously.

- Leveraging sigmoid activation and binary crossentropy loss for multi-label scenarios.

4. **Transfer Learning and Pretrained Audio Models**:
   - Fine-tuning large pretrained models such as AST (Audio Spectrogram Transformer) or BEATs on UrbanSound8K.
   - Incorporating self-supervised learning (SSL) methods to reduce dependency on labeled data.
5. **Multilingual and Multimodal Data**:
   - Applying the model to cross-lingual datasets where sound events are annotated in different languages.
   - Integrating other data modalities like video (audio-visual models) to improve contextual understanding.
6. **Robustness and Fairness**:
   - Evaluating the model across diverse demographic and geographic datasets.
   - Exploring fairness-aware training techniques to avoid bias in public safety applications.

Conclusion

In conclusion, this study advances the field of environmental sound classification (ESC) by presenting a robust, interpretable, and scalable hybrid deep learning architecture. By effectively merging the local feature extraction capabilities of Convolutional Neural Networks (CNNs) with the global temporal modeling strengths of Transformer encoders, the proposed model captures both short-term spectral details and long-range temporal dependencies inherent in non-stationary environmental audio signals.

The integration of Mel-spectrogram-based time-frequency representations, SpecAugment-based data augmentation, and a rigorous cross-validation training protocol ensures that the model not only achieves high classification accuracy but also exhibits strong generalization to unseen data. Furthermore, the inclusion of self-attention mechanisms contributes to interpretability by highlighting which temporal segments of audio the model focuses on during classification—an important step toward explainable AI in acoustic sensing.

As smart cities, autonomous systems, and IoT-enabled environments continue to evolve, the demand for real-time, reliable audio analysis systems is growing rapidly. The hybrid CNN-Transformer framework presented here provides a scalable foundation for these future applications, offering the computational efficiency of CNNs and the contextual depth of Transformers. With further adaptations, this architecture is well-positioned to serve as the core engine of next-generation audio-aware intelligent systems.

REFERENCES

[1] J. Salamon, C. Jacoby, and J. P. Bello, “A dataset and taxonomy for urban sound research,” in Proc. 22nd ACM Int. Conf. Multimedia, Orlando, FL, USA, Nov. 2014, pp. 1041–1044.

[2] A. Vaswani et al., “Attention is all you need,” in Advances in Neural Information Processing Systems (NeurIPS), Long Beach, CA, USA, 2017.

[3] A. Dosovitskiy et al., “An image is worth 16×16 words: Transformers for image recognition at scale,” in Proc. Int. Conf. Learn. Representations (ICLR), 2021.

[4] Y. Gong, Y.-A. Chung, and J. Glass, “AST: Audio Spectrogram Transformer,” in Proc. Interspeech, Brno, Czech Republic, 2021.

[5] D. S. Park et al., “SpecAugment: A simple data augmentation method for automatic speech recognition,” in Proc. Interspeech, Graz, Austria, 2019.

[6] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, “On large-batch training for deep learning: Generalization gap and sharp minima,” in Proc. Int. Conf. Learn. Representations (ICLR), 2017.

[7] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. Int. Conf. Learn. Representations (ICLR), 2015.

[8] K. Koutini, H. Eghbal-zadeh, and G. Widmer, “Efficient training of audio transformers with patchout,” in NeurIPS 2021 Workshop on Efficient Natural Language and Speech Processing, 2021.

[9] S. Chen, H. Xu, Y. Zhang, and B. Zhang, “CNN-Transformer hybrid models for environmental sound classification with SpecAugment,” IEEE Access, vol. 10, pp. 40512–40521, 2022.

[10] S. Hershey et al., “CNN architectures for large-scale audio classification,” in Proc. IEEE Int. Conf. Acoust., Speech and Signal Process. (ICASSP), New Orleans, LA, USA, Mar. 2017, pp. 131–135.

[11] K. Palanisamy, D. Singhania, and Y. Yao, “Rethinking CNN models for audio classification,” in Proc. IEEE Int. Conf. Acoust., Speech and Signal Process. (ICASSP), 2020.

[12] A. Mesaros, T. Heittola, and T. Virtanen, “A multi-device dataset for urban acoustic scene classification,” IEEE Journal of Selected Topics in Signal Processing, vol. 13, no. 1, pp. 34–48, Mar. 2019.

[13] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, “Convolutional gated recurrent neural network incorporating spatial features for audio tagging,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, no. 12, pp. 2392–2401, Dec. 2017.

[14] Q. Kong et al., “PANNs: Large-scale pretrained audio neural networks for audio pattern recognition,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 28, pp. 2880–2894, Sep. 2020.

[15] C. Wu, C. Zhang, and Y. Wang, “Sound event classification using attention-based convolutional neural networks,” IEEE Access, vol. 7, pp. 93258–93268, 2019.

[16] S. Mun, J. Lee, and H. Ko, “Deep neural network based real-time urban sound classification model,” Applied Sciences, vol. 11, no. 3, p. 1326, 2021.

[17] H. Xie, Z. Sun, X. Xie, and J. Xie, “A hybrid deep learning model with attention mechanism for environmental sound classification,” Journal of Ambient Intelligence and Humanized Computing, vol. 13, pp. 4561–4573, 2022.

18% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 8 words)

Match Groups

147 Not Cited or Quoted 18%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

10% Internet sources
9% Publications
15% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.