



Degree Project in Computer Science and Engineering

First Cycle, 15 credits

Environmental Sound Classification Using Deep Learning

Investigation of natural, human and urban environment sounds

OLIVER YOUNESS

VICTOR BORG

Environmental Sound Classification Using Deep Learning

OLIVER YOUNESS
VICTOR BORG

Degree Project in Computer Science and Engineering, First Cycle 15 credits
Date: May 26, 2026
Supervisor: Erik Fransén
Examiner: Prof. Pawel Herman
Supervisor: Erik Fransén
School of Electrical Engineering and Computer Science
Swedish title: Miljöljud-klassificering med hjälp av djupinlärning

Abstract

Environmental sound classification is a field within machine learning and signal processing focused on identifying and categorizing sounds from natural, human and urban environments. This thesis researches the development, evaluation and deployment of a deep learning-based system for classifying environmental sounds and presenting the results visually.

We employ a supervised learning approach using the publicly available ESC-50 dataset of labeled environmental audio recordings. We construct a 4-layer convolutional neural network (CNN) trained on Mel-spectrogram representations of audio as proposed for environmental sound classification. Performance is evaluated using accuracy, precision, recall, F1-score and confusion matrices to demonstrate classification results.

The results confirm that deep learning models can achieve environmental sound classification with a mean accuracy of 69.4% and that visual user feedback can be combined with real-time operations required for audio processing in a practical application. Beyond the evaluation, which included detailed analysis of precision, recall and F1-scores, the trained model was converted to ONNX format and integrated into a desktop application.

Keywords

Environmental sound classification, audio classification, deep learning, convolutional neural networks, Mel-spectrograms, audio signal processing, real-time audio processing, ESC-50 dataset, ONNX Runtime, real-time systems.

Sammanfattning

Miljöljudsklassificering är ett område inom maskininlärning och signalbehandling fokuserat på att identifiera och kategorisera ljud från naturliga, mänskliga och urbana miljöer. Detta examensarbete forskar om utvecklingen, utvärderingen och implementeringen av ett djupinlärningsbaserat system för att klassificera miljöljud och presentera resultaten visuellt.

Vi använder en övervakad inlärningsmetod med hjälp av den offentligt tillgängliga ESC-50-datan av märkta miljöljudinspelningar. Vi konstruerar ett 4-lagers konvolutionellt neuralt nätverk (CNN) tränat på Mel-spektrogramrepresentationer av ljud som föreslagits för miljöljudsklassificering. Prestanda utvärderas med hjälp av matriser för noggrannhet, precision, återkallelse, F1-poäng och förvirring för att visa klassificeringsresultat.

Resultaten bekräftar att djupinlärningsmodeller kan uppnå miljöljudsklassificering med en medelnoggrannhet på 69,4 % och att visuell användarfeedback kan kombineras med realtidsoperationer som krävs för ljudbehandling i en praktisk tillämpning. Utöver utvärderingen, som inkluderade detaljerad analys av precision, återkallelse (recall) och F1-poäng, konverterades den tränade modellen till ONNX-format och integrerades i en skrivbordsapplikation.

Nyckelord

Miljöljudsklassificering, ljudklassificering, djupinlärning, konvolutionella neurala nätverk, Mel-spektrogram, signalbehandling, realtidsljudbehandling, ESC-50-datasetet, ONNX Runtime, realtidssystem.

Acknowledgments

We would like to thank our supervisor, Erik Fransén, for valuable guidance, constructive feedback and support. We also thank KTH Royal Institute of Technology for providing the academic environment and resources necessary to conduct this work.

Contents

1. Introduction	5
1.1 Context and Motivation	5
1.2 Research Problem and Question	6
1.3 Aims and Scope	6
1.4 Thesis Outline	7
2. Background	8
2.1 History and Related Work	8
2.2 Fundamentals of Signal Processing	9
2.2.1 Signals and Frequencies	9
2.2.2 The Convolution Theorem	10
2.3 CNN and ESCNet used for ESC	11
3. Methods	13
3.1 Data Collection and Preparation	13
3.2 Preprocessing and Feature Representation	14
3.3 Model Architecture	15
3.4 Training Procedure	17
3.4.1 Loss function and Optimizer	18
3.4.2 Regularisation and Early Stopping	19
3.5 Integration into a desktop application.	20
3.5.1 Model Export and Optimization	20
3.5.2 Audio Capture and Buffering	20
3.5.3 Signal Processing Method	21
3.5.4 Inference Thread	21
3.5.5 Graphical User Interface	21
3.6 Evaluation Metrics	21
3.6.1 Classification Metrics	21
3.6.2 Testing Real-time Performance	22
4. Results	23
4.1 Classification Performance Benchmarks	23
4.2 Error Analysis	24
4.3 User Interface and Visual Feedback	25
5. Discussion	28
5.1 Comparison To Previous Research	29
5.2 Limitations	30
5.3 Ethical Considerations	30
5.4 Societal Impact	31
5.5 Scientific Contribution	31
5.6 Sustainability Considerations	32
5.7 AI Usage In This Paper	32
6. Conclusions	33
6.1 Future Work	34
References	35

Chapter 1

Introduction

1.1 Context and Motivation

Sound contains a large amount of information about the surrounding environment. The automatic interpretation of environmental sounds, commonly referred to as environmental sound classification (ESC), has become an important research area within machine learning and artificial intelligence. Unlike speech recognition, which benefits from structured linguistic patterns, environmental sounds vary significantly both within and across categories [1]. Sounds such as rain, engines, footsteps, alarms, or animal vocalizations can differ in duration, frequency content, recording quality and background noise, making classification difficult for traditional pattern recognition methods [2].

Environmental sound classification (ESC) has applications in several domains, particularly within smart city and ambient intelligence systems, including security, surveillance, manufacturing, autonomous vehicles and noise mitigation [3]. In smart home environments, ESC systems can detect alarms, doorbells, or glass breaking events to enhance safety and automation. In healthcare and assistive technology, sound classification can improve environmental awareness for individuals with hearing impairments by converting important acoustic events into visual or haptic feedback, supporting ambient assisted living applications. ESC is also widely used in urban monitoring systems for analyzing traffic noise, public environments and noise pollution patterns [3]. In industrial settings, audio-based classification enables fault detection and predictive maintenance by identifying abnormal machine sounds before failures occur [4]. More recently, autonomous systems and robots have begun integrating environmental audio analysis to improve situational awareness and complement visual sensing in complex environments [5].

Sound recognition technologies can also solve real-world accessibility problems. For individuals with hearing disability, the inability to hear important environmental sounds can create both safety concerns and communication difficulties. Systems capable of recognizing alarms, sirens, doorbells, or approaching vehicles and converting them into visual feedback could improve environmental awareness and accessibility in different situations.

This thesis investigates the development of a deep learning based environmental sound classification system together with a real-time desktop application for audio visualization and feedback. The project combines machine learning, signal

processing and software engineering to study how environmental sound classification can work as a complete real-time system.

1.2 Research Problem and Question

This project focuses on both sound classification and practical system integration by exploring how deep learning can be used within a complete interactive application. The work is based around the following research questions:

1. How well can a convolutional neural network classify environmental sounds using Mel-spectrogram representations?
2. Can the trained model be integrated into a desktop application that operates in real time without major latency or performance issues?
3. Which types of environmental sounds are the most difficult for the model to classify and what patterns appear in these classification errors?

These questions are explored through the process of designing, training, evaluating and deploying a complete environmental sound classification system.

1.3 Aims and Scope

The goal of this thesis is to design, implement and evaluate an end-to-end system for environmental sound classification with integrated visual feedback. The project combines audio preprocessing, machine learning, real-time inference and desktop application development into a single system.

The specific aims of the project are the following:

Dataset Preparation: To acquire and preprocess a suitable environmental sound dataset, specifically the ESC-50 dataset, including normalization and augmentation of audio samples.

Feature Representation: To generate Mel-spectrogram representations of environmental audio suitable for convolutional neural network training.

Neural Network Design: To design and evaluate a convolutional neural network architecture capable of learning time-frequency patterns from environmental sounds while remaining computationally efficient.

Performance Evaluation: To evaluate the trained model using metrics such as accuracy, precision, recall, F1-score and confusion matrices in order to analyze classification performance and common misclassification patterns.

Integration: To integrate the trained model into a desktop application capable of real-time audio processing and inference using ONNX Runtime.

Visual Feedback Design: To design and implement a graphical interface that presents classification results together with visual audio representations such as waveforms, spectrograms and confidence scores.

The scope of this project is limited to these goals to ensure a reasonable workload within the timeframe of the thesis. The work focuses only on supervised learning methods trained on publicly available datasets and does not include unsupervised or self-supervised learning methods. Although the system will work in real-time, no formal user studies were conducted to evaluate the effectiveness of the visual feedback for hearing-impaired users. In addition, extensive real-world deployment and clinical validation are outside the scope of this work.

1.4 Thesis Outline

Chapter 2 presents the theoretical background and related research relevant to the project, including signal processing, environmental sound classification, deep learning methods for audio analysis and real-time audio systems.

Chapter 3 describes the methodology used throughout the project. This includes dataset preparation, preprocessing and feature extraction, neural network architecture design, training procedures, evaluation methods and integration into the desktop application.

Chapter 4 presents the experimental results, including classification performance metrics, error analysis, confusion matrices and observations related to real-time system behaviour and the graphical interface.

Chapter 5 discusses the results in relation to the research questions, limitations of the system, practical deployment considerations and future improvements.

Chapter 6 concludes the thesis by summarizing the main findings and contributions of the project together with potential future work.

Chapter 2

Background

Environmental sound classification has become more important in recent years due to the higher demand for intelligent systems that can interpret surrounding environments automatically [6].

Many situations in real life rely on awareness from hearing, including emergency alarms, traffic signals, machiner warnings and human activity sounds. For individuals with hearing disabilities, important environmental sounds may go unnoticed, affecting both safety and independence [7]. Sound recognition systems can help address this issue by converting audio information into visual feedback that allows users to remain more aware of their surroundings.

Beyond assistive technology, environmental sound classification also has practical applications in industrial monitoring, smart homes, surveillance systems, robotics and public safety [6]. As machine learning and deep learning methods have advanced, convolutional neural networks have become effective for these tasks because they are able to learn complex patterns in sound directly from spectrogram representations of audio [7]. However, environmental sound classification remains difficult because of overlapping sounds, background noise and large variation that exists within many sound categories [6]. These challenges make it important to develop systems that are not only accurate in controlled datasets, but also efficient enough to operate reliably in real-time environments.

2.1 History and Related Work

Earlier ESC systems relied on feature extraction by hand combined with traditional machine learning algorithms. Common approaches involved extracting acoustic representations such as *Mel-Frequency Cepstral Coefficients* (MFCCs) or spectrogram-based features from short audio sounds and then training classifiers such as Support Vector Machines (SVMs), Hidden Markov Models (HMMs), or Random Forests [8]. These methods did achieve reasonable results on smaller datasets, however, they often struggled to generalize to more diverse and noisy real-world sounds.

The development of deep learning significantly changed this field. *Convolutional neural networks* (CNNs), which became successful in image recognition, were later adapted for audio processing tasks [9].

One of the most influential benchmark datasets introduced for this research area was ESC-50, proposed by Piczak [8], which provided a dataset containing 50 environmental sound categories.

Following the introduction of ESC-50, several CNN-based architectures showed large improvements over traditional machine learning methods. Piczak [9] showed that two-dimensional CNNs operating on spectrogram representations could successfully classify environmental sounds using convolution and pooling operations similar to those used in computer vision tasks. Salamon and Bello [10] later demonstrated that data augmentation methods such as pitch shifting and time stretching could significantly improve CNN performance on environmental sound datasets by reducing overfitting and increasing robustness to sound variation.

Even though raw waveform training approaches can achieve strong performance, they generally require larger datasets and higher computational resources compared to spectrogram-based CNN systems. Other studies have investigated hybrid CNN-Recurrent Neural Network (RNN) architectures because of the need to extract both spectral and temporal features within environmental audio [6].

2.2 Fundamentals of Signal Processing

2.2.1 Signals and Frequencies

Sound is in reality pressure that changes through something like air [8]. Because these changes or signals are continuous in nature, they must be transformed into a discrete numerical form that a digital system can interpret and manipulate [8]. This critical bridge is built through a process known as *sampling* [11]. During sampling, the amplitude of the continuous sound signal is measured at regular intervals, resulting in a sequence of discrete values that represent the audio's progression over time [8]. The precision of this digital reconstruction depends heavily on the *sampling frequency*, or the number of measurements taken per second. This rate is the highest frequency that can be accurately represented without distortion, a principle defined by the *Nyquist Shannon criterion* [9]. In practical applications, various standard frequencies tailored to specific needs: 8 kHz is common for telephony where bandwidth is limited to approximately 4 kHz , 44.1 kHz serves as the standard for CD-quality audio and rates of 96 kHz or higher are reserved for professional high resolution audio production [10], [12].

While this raw digital waveform contains the entirety of the recorded information, it is generally not suited for direct use in machine learning models [13]. The main problem is the dimensionality of the data, just a few seconds of audio recorded at 44.1 kHz translates into over a hundred thousand individual data points [13]. Raw waveforms are also very unstable, small changes in timing or slight changes in

volume can change the underlying numerical values, even if the actual sound perceived by a listener remains practically the same [13]. To solve these problems, researchers utilize alternative representations designed to highlight perceptually significant features while filtering out irrelevant variations [13].

Modern audio methodology draws inspiration from this setup, mostly through the use of the *Short-Time Fourier Transform (STFT)* [11]. Instead of attempting to analyze an entire signal at once, the system breaks the audio into shorter overlapping time windows lasting tens or hundreds of milliseconds. By applying the Fourier transform to each individual window, a *spectrogram* is produced, a visual representation that maps the signal's energy across various frequencies over time. In many classification contexts, only the magnitude or "power" of the spectrogram is retained, as phase information is often considered less important to the identification of the sounds [13].

Humans can hear higher resolution at lower frequencies and lower resolution at higher frequencies, the frequency axis is often warped using the *Mel-scale*, which is approximately logarithmic above 500 Hz [13]. By processing the spectrum through a bank of triangular filters in this Mel-frequency domain, it creates a *Mel-spectrogram* [13]. This representation is not only more meaningful but has also proven much more effective for complex sound classification tasks [13].

2.2.2 The Convolution Theorem

Convolution is a mathematical operation used to analyze how one signal interacts with another. In signal processing, convolution is commonly used to apply filters to signals in order to enhance, suppress, or detect specific patterns. For example, convolution can be used to reduce noise, or identify repeating patterns within an audio signal [11].

The idea behind convolution is that a smaller function, called a *filter* or *kernel*, is moved step-by-step across a larger signal. At each position, the values of the filter are multiplied with the corresponding values of the signal and the results are summed together. This produces a new output signal that highlights features matching the filter pattern.

For discrete signals, convolution can be mathematically expressed as:

$$y(n) = \sum_{k=-\infty}^{\infty} x[k]h[n - k]$$

where $x[k]$ represents the input signal, $h[n - k]$ represents the filter or kernel and $y[n]$ is the resulting output signal after convolution [11].

In audio processing, convolution is often applied to spectrograms or other time-frequency representations. Since spectrograms contain both temporal and frequency information, convolution allows systems to detect sound patterns across neighboring time intervals and frequency regions.

2.3 CNN and ESCNet used for ESC

Convolutional neural networks (CNNs) are widely used in image recognition, pattern analysis and audio classification tasks [9]. In *environmental sound classification* (ESC), CNNs are effective because they can learn meaningful patterns directly from spectrogram representations without relying entirely on manually picked features [8], [9].

In audio classification, CNNs commonly operate on two-dimensional spectrogram inputs. A two-dimensional convolution layer processes both the time and frequency dimensions simultaneously, allowing the network to learn relationships between neighboring regions in the spectrogram. This enables the model to identify characteristic acoustic patterns that occur across different frequencies and time intervals [9], [10]. Since many environmental sounds contain both temporal and spectral structures, two-dimensional convolutions are well suited for ESC tasks.

Another commonly used component is batch normalization. Batch normalization stabilizes the training process by normalizing intermediate activations between layers. This improves gradient flow during optimization and can reduce sensitivity to parameter initialization. Batch normalization also makes training more stable on smaller datasets such as ESC-50.

To reduce overfitting, CNN architectures often use dropout layers [9], [10]. Overfitting occurs when a model learns the training data too closely and fails to generalize well to unseen examples. Dropout randomly disables a portion of neurons during training, forcing the network to learn more robust feature representations instead of relying too heavily on individual activations. Previous ESC research has shown that dropout and data augmentation can improve classification performance on smaller environmental sound datasets [10].

CNN-based ESC systems usually contain several convolutional layers that follow sequentially [9]. Earlier layers tend to learn simpler patterns such as spectral transitions or short frequency structures, while later layers learn more abstract representations. In environmental sound classification, deeper layers may capture

more recognizable sound characteristics associated with classes such as sirens, rain, footsteps, or engine noise.

ESCNet follows the general principles used in previous CNN-based ESC research by using convolutional layers to Mel-spectrogram representations of environmental audio [9], [10]. The architecture consists of multiple convolutional blocks containing convolution layers, batch normalization, ReLU activation functions and max pooling. Together, these layers progressively extract more and more informative feature representations from the spectrogram input before the final classification.

Chapter 3

Methods

This chapter presents the methods used to develop the sound classifier. The approach is meant to be independent of any sound source, allowing the same system to be applied to many datasets like urban sounds, instruments, animal sounds etc. The methods cover data collection, preprocessing, model design, training and integration into a desktop application.

3.1 Data Collection and Preparation

For environmental sound classification, several publicly available datasets exist, each with unique characteristics. The UrbanSound8K dataset [14] contains 8732 labeled data points from ten sound classes (air conditioner, car horn, children playing, dog bark, drilling, engine idling, gunshot, jackhammer, siren, street music). The ESC-50 dataset [9] has 2000 recordings for 50 balanced classes, including both environmental and speech sounds. For larger tasks, AudioSet [9] provides over two million clips annotated with 632 sound classes. However, it is not optimal considering its scale and label noise. In this project, the ESC-50 dataset was chosen because it provided the necessary middle ground between complexity and manageability.

The primary advantage of using ESC-50 is its balance with exactly 40 examples per class, the dataset does not have a long-tailed distribution and class imbalance like other datasets. This structure makes the training process simpler, because it removes the need for complex handling of over-represented categories.

Despite its balance, 2000 samples is a relatively small size for training deep learning models. To address this and improve generalization, data augmentation techniques, such as time-stretching and pitch-shifting, to artificially increase the variety of our samples without altering the underlying sound content, was applied. The final dataset was partitioned into training (80%) and validation (20%).

No.	Major Categories				
	Animals	Natural Soundscapes and Water Sounds	Human, Non-Speech Sounds	Interior/Domestic Sounds	Exterior/Urban Noises
1	Dog	Rain	Crying baby	Door knock	Helicopter
2	Rooster	Sea waves	Sneezing	Mouse click	Chainsaw
3	Pig	Crackling fire	Clapping	Keyboard typing	Siren
4	Cow	Crickets	Breathing	Door, wood creaks	Car horn
5	Frog	Chirping birds	Coughing	Can opening	Engine
6	Cat	Water drops	Footsteps	Washing machine	Train
7	Hen	Wind	Laughing	Vacuum cleaner	Church bells
8	Insects (flying)	Pouring water	Brushing teeth	Clock alarm	Airplane
9	Sheep	Toilet flush	Snoring	Clock tick	Fireworks
10	Crow	Thunderstorm	Drinking, sipping	Glass breaking	Hand saw

Figure 1. Categories and classes of the ESC-50 dataset [15]

3.2 Preprocessing and Feature Representation

Raw audio signals contain a large amount of temporal information, therefore, the recordings

were transformed into Mel-spectrogram representations before being used by the neural

network. Mel-spectrograms produce a time-frequency representation that aligns better with human hearing perception while also reducing the dimensionality of the signal.

All audio clips were converted to mono format and normalized before preprocessing. The Librosa library was used to compute Mel-spectrograms with 128 Mel frequency bins using a Fast Fourier Transform (FFT) window size of 2048 samples and a hop length of 512 samples. The maximum frequency was limited to 8000 *Hz* to focus on the most relevant frequency range for environmental sounds.

After generation, the Mel-spectrograms were normalized to the range [0, 1] and resized into a fixed input dimension of 128×216 . These spectrogram representations were then used as input to the convolutional neural network.

To improve robustness and generalization during training, data augmentation techniques were applied dynamically to the audio clips before spectrogram generation. These included pitch shifting, time stretching, random temporal shifting and low-level Gaussian noise Injection.

3.3 Model Architecture

The environmental sound classifier was implemented as a four-block two-dimensional convolutional neural network (2D CNN) called ESCNet. The model was designed to process Mel-spectrogram inputs with the shape (1, 128, 216), where the dimensions represent the Mel-frequency bins and temporal frames. Since Mel-spectrograms encode both frequency and temporal information, they can be treated similarly to grayscale images, making convolutional neural networks well suited for the task.

The overall goal of the architecture was to balance classification performance with computational efficiency. Because the ESC-50 dataset is relatively small, the network needed to be lightweight enough to reduce overfitting while still being capable of learning meaningful acoustic patterns from the data. The architecture was therefore designed around standard convolutional building blocks together with parameter-efficient operations such as global average pooling.

Each block of the network contains:

- a two-dimensional convolution layer,
- batch normalization,
- ReLU activation,
- and 2×2 max pooling.

The convolutional layers use 3×3 kernels with padding set to one. Using small kernels is a common design choice in CNN architectures because it allows the model to gradually learn local spatial patterns while keeping the number of trainable parameters relatively low. In the context of environmental sound classification.

After convolution operations, activation functions are applied to introduce nonlinearity into the network. One of the most commonly used activation functions is the Rectified Linear Unit (ReLU) [9]. ReLU outputs the input value when it is positive and outputs zero otherwise. This can be written as:

$$f(x) = \max(0, x)$$

Pooling layers are typically placed after convolution and activation layers in order to reduce the spatial dimensions of feature maps [9]. A common approach is max pooling, where only the maximum value within a small region is retained. In many CNN architectures, 2×2 max pooling is used to progressively reduce feature map

size while preserving the strongest activations. Pooling reduces computational cost and helps the network become less sensitive to small variations in the input [12].

Batch normalization is applied after every convolution to stabilize training and improve gradient flow throughout the network. This becomes important when training on relatively small datasets such as ESC-50, where unstable optimization can lead to overfitting or poor convergence. ReLU activation functions are used to allow non-linearity while remaining computationally efficient.

Each convolution block is followed by 2×2 max pooling, which gradually reduces the spatial dimensions of the feature maps while preserving the strongest activations. Pooling also reduces computational cost and increases the receptive field of deeper layers, allowing the network to capture more abstract acoustic structures at later stages of the architecture.

The number of feature channels increases progressively through the network:

- Block 1: 32 channels
- Block 2: 64 channels
- Block 3: 128 channels
- Block 4: 128 channels

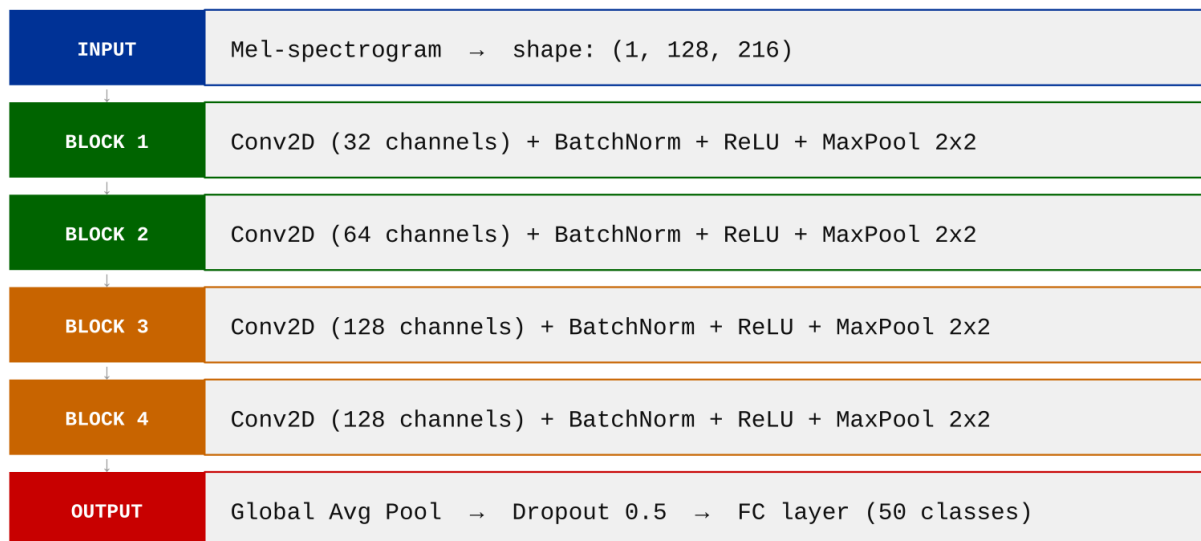


Figure 2. The layers of the CNN model (called ESCNet).

This gradual increase allows the network to learn more complex feature representations by each step as the data moves deeper through the architecture. Earlier layers tend to capture simpler local structures such as edges, or short

frequency patterns, while deeper layers learn more complex combinations of acoustic features that are more specific to particular sound categories.

After the final convolutional block, global average pooling (GAP) is applied to reduce the remaining spatial dimensions into a single 128-dimensional feature vector. Instead of flattening the entire feature map into a large layer, GAP computes the average activation value of each feature channel. This greatly reduces the number of trainable parameters and lowers the risk of overfitting.

The use of global average pooling was an important design decision in this project. Earlier versions of the architecture used flattening followed by large fully connected layers, which resulted in a very high parameter count and bad generalization performance on the small dataset. By replacing this structure with GAP, the classifier head became much more compact while still preserving the most important learned information from the convolutional layers.

After global average pooling, a dropout layer with probability 0.5 is applied before the final classification layer. Dropout randomly disables part of the feature representation during training, forcing the network to learn more robust and distributed representations instead of relying too heavily on specific activations.

The final fully connected layer maps the 128-dimensional representation to 50 output classes corresponding to the ESC-50 categories. The network outputs logits for each class, which are then converted into probabilities during evaluation using the softmax function.

3.4 Training Procedure

Training deep neural networks on relatively small environmental sound datasets introduces several challenges related to overfitting, optimization stability and generalization. Because the ESC-50 dataset contains only 2000 recordings distributed across 50 sound classes, the model can easily memorize training examples instead of learning more general acoustic patterns. For this reason, careful training procedures and regularization strategies were required throughout the project.

The model was trained using supervised learning, where each audio clip is paired with a ground-truth class label. During training, batches of Mel-spectrogram representations were passed through the network, producing predicted class probabilities for the 50 ESC-50 categories. The predictions were then compared against the true labels using a loss function and the network weights were updated through backpropagation using gradient descent optimization.

To evaluate generalization performance reliably, the official five-fold cross-validation split provided with the ESC-50 dataset was used throughout the experiments. In each fold, four subsets of the dataset were used for training while the remaining subset was reserved for validation. This process was repeated five times so that every recording was used once for validation. The final reported performance represents the average classification accuracy across all five folds.

Using cross-validation was important because the dataset is relatively small. A single train-test split could produce unstable or misleading performance estimates depending on which recordings happened to be included in the validation set. Cross-validation instead provides a more robust estimate of how well the model generalizes across different subsets of environmental sounds.

During training, the Mel-spectrogram representations were generated dynamically together with data augmentation operations. Instead of storing large numbers of augmented audio files on disk, transformations were applied during runtime before feature extraction. This increased variation in the training data while reducing storage requirements.

The augmentation pipeline included:

- random pitch shifting,
- time stretching,
- temporal shifting,
- and additive Gaussian noise.

Pitch shifting slightly modifies the frequency content of the audio without changing the overall semantic category, while time stretching alters the playback speed without strongly affecting pitch. Temporal shifting changes the starting position of the sound within the audio clip, making the model less dependent on exact temporal alignment. Gaussian noise injection helps improve robustness against noisy real-world recording conditions.

These augmentations were especially important because environmental sounds often vary naturally in recording conditions, intensity and timing. Increasing training variability helped the model learn more stable time-frequency patterns instead of memorizing specific samples.

3.4.1 Loss function and Optimizer

The model was trained using cross-entropy loss. This loss function works by looking at the raw scores the network produces for each class and measuring how far off the prediction is from the correct answer. The further the network is from the right class, the higher the loss and by minimizing this during training, the network gradually learns to assign higher confidence to the correct sound category.

Optimization was performed using the Adam optimizer [16], which combines momentum and adaptive learning rate scaling to improve convergence stability. Adam was selected because it performs well across a wide range of deep learning tasks and generally requires less manual tuning compared to traditional stochastic gradient descent methods.

The optimizer was configured with:

- an initial learning rate of 0.001,
- $\beta_1 = 0.9$,
- and $\beta_2 = 0.999$.

β_1 handles momentum. It averages past updates to keep the model moving in a smooth, consistent direction, preventing it from overreacting to small, noisy changes. β_2 handles scaling. It tracks how much each weight changes and automatically adjusts the speed. These values follow commonly used defaults for Adam optimization and provided stable training behavior throughout the experiments. Training was performed independently for each fold during cross-validation. At the end of each epoch, validation loss and validation accuracy were evaluated on the held-out fold. The model weights corresponding to the best validation performance were saved and later used for evaluation.

3.4.2 Regularisation and Early Stopping

Several regularization techniques were used in order to reduce overfitting and improve generalization on the relatively small ESC-50 dataset.

One important regularization technique was dropout. A dropout layer with probability 0.5 was applied after the global average pooling layer and before the final classification layer. During training, dropout randomly disables part of the feature representation, preventing the network from becoming overly dependent on specific neurons. This encourages the model to learn more distributed and robust feature representations.

Batch normalization was also included after every convolutional layer. In addition to improving optimization stability, batch normalization acts as a mild regularizer by introducing small amounts of noise into the feature statistics during training. This helped stabilize gradient flow and reduced sensitivity to parameter initialization.

Early stopping was used to monitor validation loss during training. If the validation loss failed to improve for eight consecutive epochs, training was stopped automatically and the model weights with the lowest validation loss were restored. This prevented the model from continuing to train after beginning to overfit the training data.

In addition, a learning rate scheduler was used to reduce the learning rate when validation performance converged. Lower learning rates during later training stages allowed the optimizer to make smaller parameter updates and converge more smoothly toward a local minimum. Together, dropout, batch normalization, data augmentation and early stopping formed the main regularization strategy used throughout the project.

3.5 Integration into a desktop application.

As part of the goal, the trained model was integrated into a desktop application written in C++. The system combines audio input, signal processing, machine learning functionality and graphical rendering into one pipeline. Audio is recorded either from a microphone or system output and stored continuously in a ring buffer. This audio data is then converted into a Mel-spectrogram, which is used as input to the model. At the same time, the application displays visual elements such as an FFT spectrum, spectrogram and the predicted sound class.

Implementing machine learning models in real-time systems introduces several technical challenges. Audio processing must operate within strict timing constraints to avoid interruptions, glitches or delays in the audio stream. Since incoming audio data arrives continuously, the system must process and classify sound segments quickly enough to keep up with the input.

3.5.1 Model Export and Optimization

To use the trained model in the application, it was exported to ONNX format and executed using the ONNX Runtime library. This allows models trained in Python to run directly in a C++ environment. The system was designed so that different model versions can be loaded using the same interface. This makes it easy to test different models during development and later replace them without changing the rest of the application.

3.5.2 Audio Capture and Buffering

Audio input is handled using the PortAudio library, which gives access to available audio devices on the system. Audio samples are continuously written into a fixed-size ring buffer that always contains the most recent audio data. When the buffer becomes full, older samples are replaced by new ones. This keeps memory usage stable while ensuring that the system always works with new audio.

3.5.3 Signal Processing Method

The input audio is transformed into a Mel-spectrogram through a signal processing pipeline. First, the audio signal is divided into small overlapping segments and processed using a Fast Fourier Transform to extract frequency information. A power spectrum is then computed and mapped to the Mel scale using a filter. After that, the values are converted to a logarithmic scale to better match human perception of sound. The resulting Mel-spectrogram is used as input to the model. Additional processing is also performed to generate waveform and frequency spectrum data for visualization in the interface.

3.5.4 Inference Thread

Model inference is performed in a separate background thread. At regular intervals of around half a second, the thread reads the latest audio segment from the ring buffer and converts it into a Mel-spectrogram. The model processes this input and outputs probabilities for each sound class. The class with the highest probability is selected as the prediction, along with a confidence score. The results are stored in a shared data structure protected by a mutex and a callback is used to notify the main thread when new predictions are available. Running inference separately ensures that it does not interfere with audio capture or affect real-time performance.

3.5.5 Graphical User Interface

The graphical interface was implemented using OpenGL. The application runs a main loop that continuously updates the display at the screen refresh rate. Each frame, it checks for new inference results and updates the visual output accordingly. The interface shows the predicted sound class, confidence level and additional visualizations such as spectrum. Because rendering is handled separately from audio processing and inference, the interface remains smooth and responsive during real-time operation.

3.6 Evaluation Metrics

3.6.1 Classification Metrics

The performance of the model was evaluated using standard classification metrics. Accuracy measures how often the model predicts the correct class. To better understand performance across different sound categories, precision, recall and F1 score were calculated for each class. A confusion matrix was also used to analyze common misclassifications, especially between acoustically similar sounds.

3.6.2 Testing Real-time Performance

The system was also evaluated based on its real-time performance. Audio input was monitored to ensure that there were no interruptions or other errors. CPU and memory usage were measured to confirm that the application runs efficiently. By separating audio input, inference and rendering into different threads, the system maintained stable performance while continuously processing and visualizing audio in real time.

Chapter 4

Results

4.1 Classification Performance Benchmarks

To evaluate the performance of the proposed ESCNet model, several classification tests were conducted using the ESC-50 dataset together with five-fold cross-validation. The purpose of these tests was to answer the main research questions regarding how well convolutional neural networks can classify environmental sounds and whether the learned representations generalize across different subsets of data.

The evaluation focused on four main metrics: accuracy, precision, recall and F1-score. Accuracy measures the overall percentage of correctly classified sounds and provides a general overview of model performance. Precision and recall were used to better understand how the classifier behaves for individual sound categories. Precision measures how many predicted samples of a class are actually correct, while recall measures how many real samples of a class are successfully detected by the model. The F1-score combines both precision and recall into a single metric and is especially useful when evaluating balanced multi-class classification tasks such as ESC-50.

The tests also aimed to evaluate how stable the model performance remained across different validation folds. Since the ESC-50 dataset is relatively small, cross-validation was important to ensure that the reported performance was not dependent on a single train-test split. Consistent performance across folds would indicate that the model learns more general acoustic features instead of memorizing specific recordings.

Metric	Value
Accuracy	69.4%
Average Precision	67%
Average recall	66%
Average F1 score	66%

Table 1. Average validation performance metrics over 5-fold cross validation.

The original Piczak CNN achieved approximately 64.5% accuracy, while more recent but still lightweight convolutional architectures typically fall in the range of 65% to 70% [9]. The ESCNet model therefore performs at the higher end of that range, despite having a relatively small number of parameters.

The balanced performance across folds suggests that the combination of Mel-spectrogram features, batch normalisation, global average pooling and data augmentation provides stable generalisation. The model does not rely on a lucky split of the data, instead, it learns meaningful acoustic patterns.

4.2 Error Analysis

In addition to the numerical metrics, confusion matrices were analyzed to investigate which sound categories were most difficult to separate. This helped answer the research question regarding which environmental sounds are commonly misclassified and whether these errors follow understandable acoustic patterns rather than occurring randomly.

The model performed best on sounds with distinctive, stable acoustic structures. For example, gunshots were correctly identified in 96% of cases and the false negative rate was almost zero. Sirens and vacuum cleaners also achieved recall above 90%. These sounds are strong noises that do not vary much between different recordings.

The most common confusions occurred between pairs of classes that share similar temporal and spectral characteristics. The following pairs showed the highest mutual misclassification rates:

Rain and thunderstorm: 17% of rain samples were predicted as thunderstorms and 14% of thunderstorm samples were predicted as rain.

Drilling and jackhammer: 15% of drilling sounds were confused with jackhammer and 13% of jackhammer sounds were confused with drilling.

Chirping birds and crickets: 12% of chirping birds were misclassified as crickets and a symmetric 11% of crickets were misclassified as birds.

Crying baby and laughing: surprisingly, despite both being human sounds, the confusion rate was only 3%. This indicates that the model is not simply learning a high-level “human voice” category. Instead, it picks up on acoustic differences.

Pigs and dogs: confusion occurred in about 9% of cases.

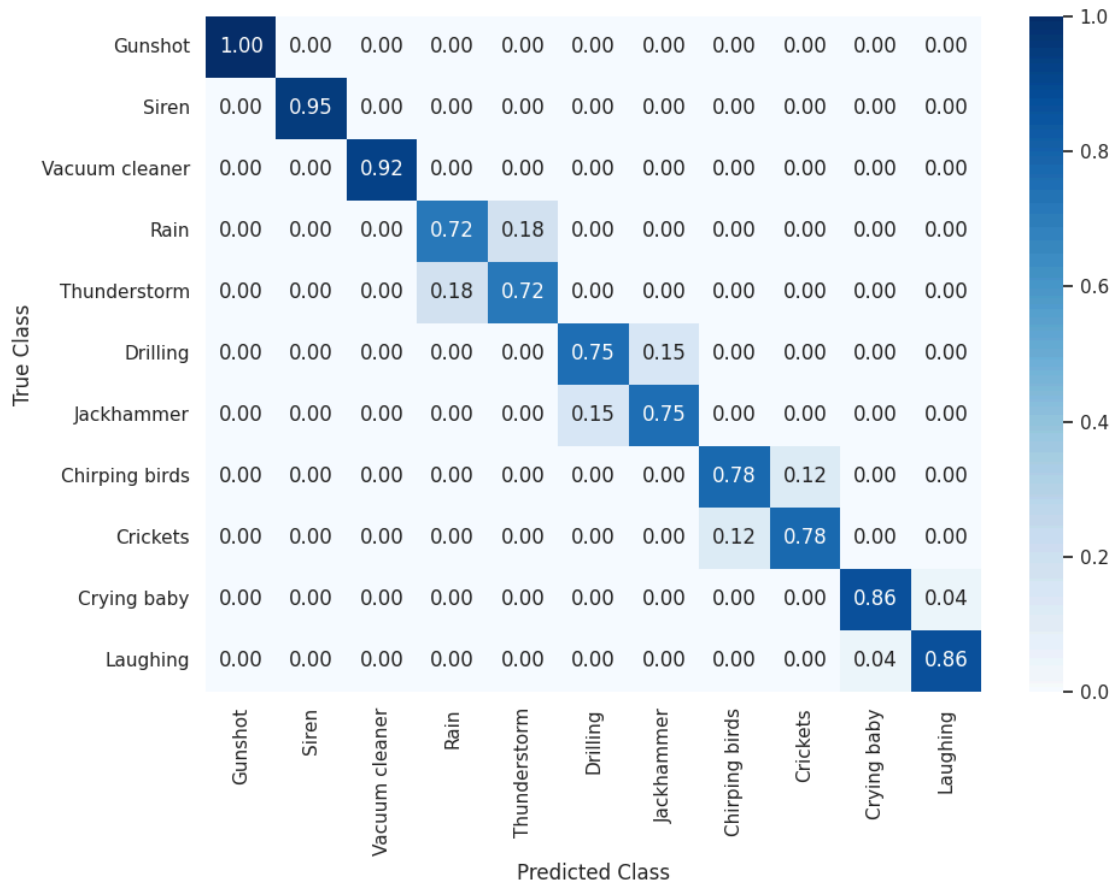


Figure 3: Confusion matrix of the classification on the validation set. The blue color scale represents the proportion of samples classified in each category.

The confusion matrix shows that errors are not random. They concentrate on acoustically similar classes, which would also be difficult for humans to distinguish from short clips. The fact that completely unrelated classes (for example, clock alarm and rain) are almost never confused suggests that the model is learning genuine acoustic features rather than exploiting dataset artefacts.

4.3 User Interface and Visual Feedback

The ring buffer design for audio capture worked without glitches or underruns, even when the inference thread occasionally took longer due to background system activity. Separating audio capture, inference and graphical rendering into different threads prevented any single component from blocking the others. The application consistently maintained a stable audio stream and responsive user interface.

The interface uses a dark theme with muted blue and gray colours for text and visual elements. There are no cluttered controls or complex menus, the user simply

launches the application and it immediately starts listening and displaying results. This minimal design helps the user focus on the sound events happening in the environment without distraction.

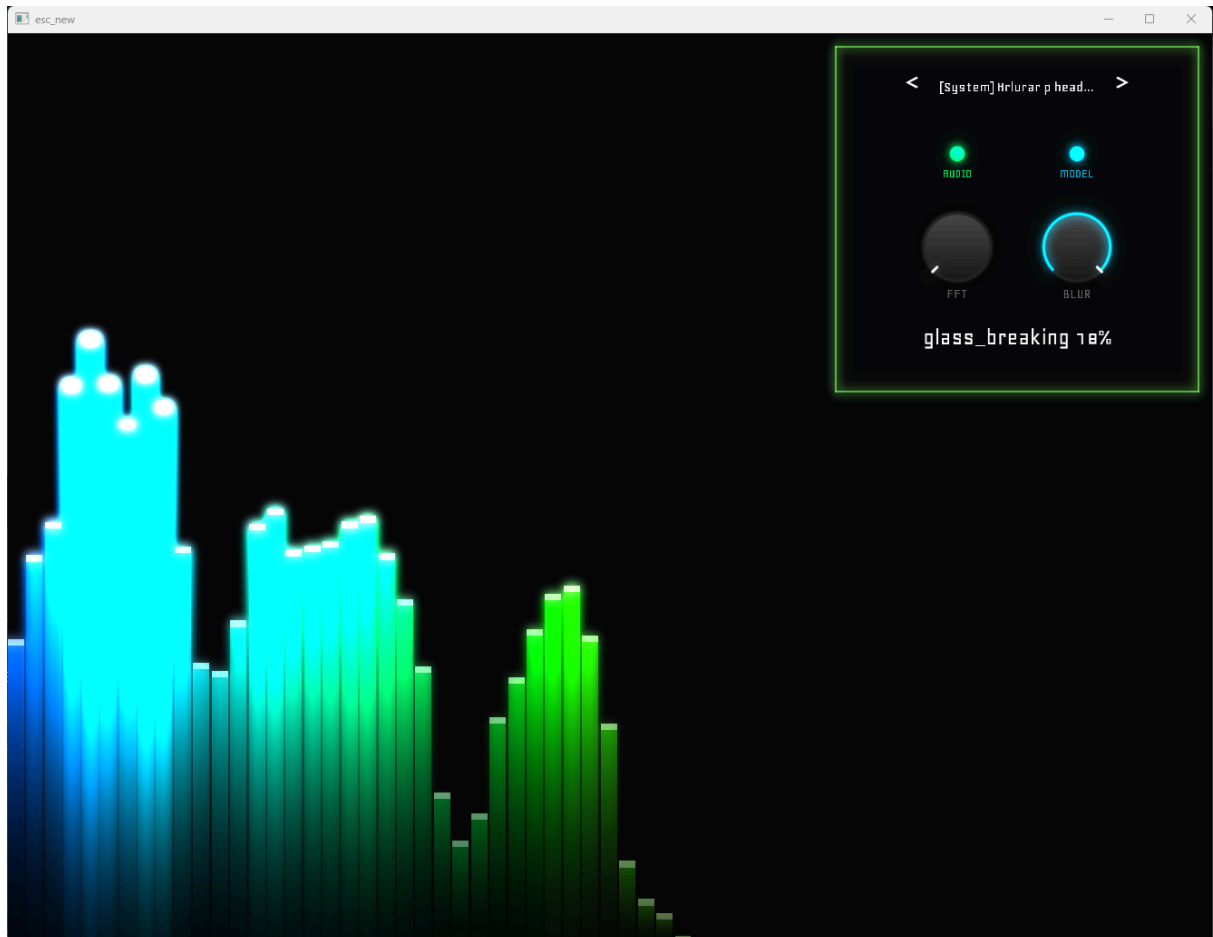


Figure 4: Interface of the application. The x-axis represents the frequency components of the audio signal (Hz), while the y-axis represents the amplitude of each frequency component, indicating the signal strength at that frequency.

The most important feature of the application, shown in figure 4, is the Fast Fourier Transform (FFT) spectrum in the background, which provides immediate visual feedback by displaying the frequency spectrum of the input signals. The colors of the FFT spectrum shown in the left bottom corner in figure 4 represent different frequency ranges, where blue indicates lower frequencies and green indicates higher frequencies.

To keep the design simple and reduce cognitive load, the controls are together in a small box in the corner. This box uses familiar icons that are easy to recognize, allowing the user to change settings like the visual smoothness or the audio input source without. Small status lights show whether the audio and the AI model are

working, which helps with error prevention and system monitoring. Finally, by showing a confidence percentage, the interface communicates the AI's "certainty" in a way that is easy for a human to interpret, building a sense of trust between the user and the software.

Chapter 5

Discussion

The experimental results show that the ESCNet model achieves a mean accuracy of 69.4% on the ESC-50 dataset. This is a solid result for a relatively small 2000 sample dataset and it suggests that Mel-spectrograms combined with a four-block CNN architecture are a suitable choice for environmental sound classification. The use of batch normalisation after every convolutional layer stabilised training and global average pooling instead of a large fully connected head drastically reduced the number of parameters, which prevented the significant overfitting.

The model performed very well on sounds with distinctive and stable time-frequency patterns, such as gunshots, sirens and vacuum cleaners. These sounds have little intra-class variation, so the model can learn a compact representation that generalises easily. At the other extreme, classes like rain, thunderstorm, drilling and jackhammer are difficult to separate even for human listeners, especially when only short five-second clips are available. The confusion between chirping birds and crickets is also understandable because both sounds consist of rapid, high-frequency pulses. The fact that the model sometimes confuses these classes is not a sign of a bad architecture, it is instead a reflection of the similarity in the sounds. Improving performance would likely require longer audio contexts, which is something to consider testing in the future of this work.

One encouraging observation is that the model rarely confused crying baby with laughing, even though both are human vocal sounds. This suggests that the model is not simply memorising categories like “human voice”. Instead, it learns to distinguish the specific acoustic features of for example crying and laughing.

The integration was successful. The ONNX Runtime library provided a way to deploy the model inside the desktop application. The multithreaded design with a ring buffer for audio capture ensured that inference never blocked the audio stream. Latency remains low in such a way that the user sees the prediction almost immediately after a sound occurs. For environmental sound awareness, a response time of half a second is acceptable because most important sounds last longer than that and very short transient sounds could potentially be missed if they fall exactly between two inference windows, but this can be mitigated by overlapping analysis windows, which is a straightforward future improvement.

5.1 Comparison To Previous Research

The results of this project can be compared with several earlier studies within environmental sound classification. One of the most influential works in this field is the CNN-based approach proposed by Piczak [9], where two-dimensional convolutional neural networks were applied to spectrogram representations of environmental sounds. Piczak reported an accuracy of approximately 64.5% on the ESC-50 dataset. The ESCNet model developed in this project achieved a mean accuracy of 69.4% [9], which is higher despite using a relatively lightweight architecture. This suggests that the combination of batch normalization, global average pooling and data augmentation contributed positively to the final performance.

The work by Salamon and Bello [10] demonstrated that data augmentation plays an important role in environmental sound classification, especially when training data is limited. Their research showed that techniques such as pitch shifting and time stretching could improve robustness and reduce overfitting in CNN-based systems. Similar observations were made in this project. Without augmentation, the model quickly memorized parts of the training data and validation accuracy became unstable. After augmentation methods were introduced, validation performance became more consistent across folds.

The goal of this project was not only to maximize classification accuracy, but also to maintain low computational complexity and support real-time deployment inside a desktop application. In this context, the achieved performance is competitive because the model remains lightweight enough for real-time inference on standard hardware.

Another important difference between this project and many previous studies is the focus on full system integration. A large part of existing ESC research focuses mainly on classification accuracy under offline experimental conditions. In contrast, this work also investigated deployment, latency, multithreaded audio processing and graphical feedback inside a complete real-time application. This practical aspect makes the project more comparable to real-world assistive or monitoring systems rather than purely theoretical machine learning benchmarks.

At the same time, the limitations observed in this project are consistent with challenges discussed throughout earlier ESC literature. Similar to previous studies, the model struggled most with acoustically similar classes and with sounds containing large variation between recordings. This suggests that environmental sound classification remains fundamentally difficult even when using modern deep learning methods. The results therefore support the general conclusion found in earlier research: convolutional neural networks are highly effective for ESC tasks,

but performance is still strongly influenced by dataset size, recording quality and class similarity.

Overall, the findings of this project align well with previous research in the field while also demonstrating that relatively lightweight CNN architectures can still achieve strong performance when combined with careful preprocessing, regularization and real-time system design.

5.2 Limitations

Despite the positive results, this work has several limitations. The first and most important limitation is the dataset. ESC-50 is small (only 2000 recordings) and perfectly balanced. Real world environments are not small and they are not balanced. Realistically, some sounds occur more often (speech, footsteps, car noise) while others are rare (glass breaking, gunshots, alarms). The model has not been trained or evaluated on imbalanced, long-duration, or overlapping sound scenarios. Therefore its performance in a genuine everyday setting may be lower than the reported 69.4%.

The second limitation is the fixed length of five-second audio data. The model can only classify sounds that fit entirely inside that window. Very short sounds (less than half a second) may be diluted by surrounding silence and very long continuous sounds (such as a running engine or a sustained alarm) may be classified repeatedly with possible flickering between classes of sound.

5.3 Ethical Considerations

Privacy concerns might be raised for systems that continuously listen to the environment. The application processes all audio locally on the user's computer. No audio data is ever saved to disk or sent over the network. This local-only design reduces the risk of unintended surveillance or data leakage. However, users should still be aware that the microphone is active and that the system is processing sound.

Reliability is another ethical issue. If the system fails to recognise an important sound such as a fire alarm or a breaking window, the user might miss a critical warning if it is used in a critical scenario. Also, false positives could cause annoyance or unnecessary anxiety. Because the model is not perfect, the application should be presented as an experimental assistive tool rather than a certified safety device. Any real-world deployment in a safety-critical context would require more serious validation.

5.4 Societal Impact

The results of this project show that environmental sound classification systems can be used in practical real-time applications, which gives it potential societal value. One important area is assistive technology for individuals with hearing impairments. The model performed especially well on distinctive warning-related sounds such as sirens and alarms, achieving very high recall on many of these categories. This suggests that systems like the one developed in this project could help users become more aware of important environmental events through visual feedback.

The integration to a desktop application also shows that this type of method for classifying sound can run continuously on standard hardware without needing expensive infrastructure. This increases accessibility and makes the system more realistic for everyday usage for individuals. Beyond this specific application, environmental sound classification could also be useful in industrial monitoring, smart home systems.

At the same time, the limitations of the results are important to consider from a societal perspective. The model still confuses acoustically similar sounds such as rain and thunderstorms or drilling and jackhammers. In a real-world situation, incorrect predictions could reduce trust in the system or cause users to miss important events. Because of this, the application should currently be considered an assistive support tool rather than a replacement for certified safety systems.

Privacy is another important societal and ethical consideration. Systems that continuously process audio may raise concerns about surveillance and personal privacy. In this project, all audio processing is performed locally on the user's device and no recordings are stored or transmitted over the internet. This reduces some of the ethical concerns related to data collection and monitoring.

5.5 Scientific Contribution

One scientific contribution of this project is the demonstration that relatively lightweight CNN architectures can still achieve great performance for environmental sound classification while remaining efficient enough for deployment. Many previous studies mainly focus on maximizing classification accuracy in offline experiments, while this work additionally investigated deployment, latency and graphical feedback in an interactive application.

The project also contributes by showing how machine learning, signal processing and software engineering can be combined into one integrated system. The successful use of ONNX Runtime and multithreaded processing demonstrates a practical system for deploying deep learning models.

5.6 Sustainability Considerations

From a sustainability perspective, the lightweight architecture is also beneficial. Larger deep learning systems often require significant computational resources and energy consumption. ESCNet was designed to remain computationally efficient while still achieving good classification performance. Since the model can run on ordinary consumer hardware without cloud-based processing, both hardware requirements and energy usage are reduced compared to larger AI systems.

5.7 AI Usage In This Paper

We used OpenAI Codex as a coding assistant to help build parts of our classification system. Specifically, it helped us write boilerplate code, speed up data preprocessing and debug syntax errors. However, Codex was only used for technical support. We reviewed and tested all the code ourselves to make sure it was accurate and fit our project. The core ideas, data analysis and final results of the classification systems are entirely our own work.

Chapter 6

Conclusions

This project demonstrated the development, implementation and evaluation of a complete end-to-end system for environmental sound classification using deep learning and real-time audio processing. The project investigated both the classification performance of a CNN based ESC system and the practical deployment of that system inside a desktop application.

The first research question was about how well a convolutional neural network can classify environmental sounds using Mel-spectrogram representations. The experimental results showed that the proposed ESCNet architecture achieved a mean accuracy of 69.4% across five-fold cross-validation on the ESC-50 dataset which suggests mel spectrograms together with CNN architectures are effective for learning meaningful acoustic patterns from environmental audio. Sounds with clear and stable frequency structures, such as gunshots, sirens and vacuum cleaners, were classified especially well, while acoustically similar classes such as rain and thunderstorms remained more difficult to separate.

The second research question investigated whether the trained model could be integrated into a desktop application capable of operating in real time without major latency or performance issues. The results showed that this was successfully achieved through the use of ONNX Runtime, multithreaded processing and a ring buffer for continuous audio capture. By separating audio processing, inference and graphical rendering into different threads, the application maintained stable performance and low latency during operation.

The third research question focused on which types of sounds were most difficult for the model to classify and what patterns appeared in these errors. The confusion matrix analysis revealed that classification errors usually occurred between classes sharing similar temporal and spectral structures. These findings suggest that the model learned genuine acoustic representations rather than relying on random dataset patterns.

Despite the positive results, several limitations remain. The ESC-50 dataset is relatively small and balanced compared to real-world sound environments and the current system only performs single-label classification on fixed five-second windows. Real environments often contain overlapping sounds, changing background noise and much larger variation than the dataset used in this work.

In conclusion, this thesis demonstrates that environmental sound classification using deep learning is both technically feasible and practically deployable within a

real-time desktop application. The project highlights the potential of combining machine learning, signal processing and software engineering to create systems capable of interpreting and visualizing environmental audio in meaningful ways.

6.1 Future Work

In the future, focus should be placed on improving both classification performance and real-world usability. One possible improvement would be to train on larger and more diverse datasets containing overlapping environmental sounds and more realistic recording conditions. Performance could also be improved further by using overlapping inference windows and additional optimization techniques to reduce latency.

Future studies should also include user testing, especially involving individuals with hearing impairments, in order to evaluate how effective the visual feedback system is in practical situations. Such evaluations would provide important insight into usability, accessibility and trust in environmental sound classification systems used as assistive technology.

References

- [1] L. Fredianelli, G. Iannace and A. Akbaba, "Spectral and tonality characterization of unwanted environmental sounds for automatic noise assessment," *Appl. Acoust.*, vol. 244, p. 111171, 2026, doi: 10.1016/j.apacoust.2025.111171.
<https://doi.org/10.1016/j.apacoust.2025.111171>

- [2] D. Meedeniya, I. Ariyaratne, M. Bandara, R. Jayasundara and C. Perera, "A Survey on Deep Learning Based Forest Environment Sound Classification at the Edge," *ACM Comput. Surv.*, vol. 56, no. 3, Art. no. 66, Mar. 2024, doi: 10.1145/3618104
<https://dl.acm.org/doi/10.1145/3618104>

- [3] A. F. R. Nogueira, H. S. Oliveira, J. J. M. Machado and J. M. R. S. Tavares, "Sound classification and processing of urban environments: A systematic literature review," *Sensors*, vol. 22, no. 22, p. 8608, 2022, doi: 10.3390/s22228608.
<https://doi.org/10.3390/s22228608>

- [4] M. Mohaimenuzzaman, C. Bergmeir, I. West and B. Meyer, "Environmental sound classification on the edge: A pipeline for deep acoustic networks on extremely resource-constrained devices," *Pattern Recognition*, vol. 133, p. 109025, 2023, doi: 10.1016/j.patcog.2022.109025.
<https://doi.org/10.1016/j.patcog.2022.109025>

- [5] F. Walden, S. Dasgupta, M. Rahman and M. Islam, "Improving the environmental perception of autonomous vehicles using deep learning-based audio classification," 2022, doi: 10.48550/arXiv.2209.04075
<https://doi.org/10.48550/arXiv.2209.04075>

- [6] A. Bansal and N. K. Garg, "Environmental Sound Classification: A descriptive review of the literature," *Intell. Syst. Appl.*, vol. 16, p. 200115, 2022, doi: 10.1016/j.iswa.2022.200115
<https://doi.org/10.1016/j.iswa.2022.200115>

[7] G. Park and S. Lee, "Environmental noise classification using convolutional neural networks with input transform for hearing aids," *International Journal of Environmental Research and Public Health*, vol. 17, no. 7, p. 2270, 2020, doi: 10.3390/ijerph17072270.

<https://doi.org/10.3390/ijerph17072270>

[8] J. Salamon and J. P. Bello, "Feature learning with deep scattering for urban sound analysis," 2015 23rd European Signal Processing Conference (EUSIPCO), Nice, France, 2015, pp. 724-728, doi: 10.1109/EUSIPCO.2015.7362478.

<https://doi-org.focus.lib.kth.se/10.1109/EUSIPCO.2015.7362478>

[9] K. J. Piczak, "Environmental sound classification with convolutional neural networks," 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), Boston, MA, USA, 2015, pp. 1-6, doi: 10.1109/MLSP.2015.7324337.

<https://doi-org.focus.lib.kth.se/10.1109/MLSP.2015.7324337>

[10] J. Salamon and J. P. Bello, "Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification," in *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279-283, March 2017, doi: 10.1109/LSP.2017.2657381.

<https://doi-org.focus.lib.kth.se/10.1109/LSP.2017.2657381>

[11] D. Sundararajan, *Digital Signal Processing: An Introduction*, 1st ed., Springer, 2021, doi: 10.1007/978-3-030-62368-5.

<https://doi.org/10.1007/978-3-030-62368-5>

[12] C. Zhang, G. Evangelopoulos, S. Voinea, L. Rosasco and T. Poggio, "A deep representation for invariance and music classification," 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 2014, pp. 6984-6988, doi: 10.1109/ICASSP.2014.6854954.

<https://doi-org.focus.lib.kth.se/10.1109/ICASSP.2014.6854954>

[13] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S.-Y. Chang and T. Sainath, "Deep learning for audio signal processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 2, pp. 206-219, May 2019, doi: 10.1109/JSTSP.2019.2908700.

<https://doi.org/10.1109/JSTSP.2019.2908700>

- [14] Salamon, J., Jacoby, C., & Bello, J. P. "A dataset and taxonomy for urban sound research," in Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, FL, USA, Nov. 2014, pp. 1041–1044. Doi: 10.1145/2647868.2655045
<https://doi.org/10.1145/2647868.2655045>
- [15] "A CNN sound classification mechanism using data augmentation - Scientific figure," *ResearchGate*. [Online]. Available: https://www.researchgate.net/figure/The-major-categories-and-semantical-classes-of-the-ESC-50-dataset_tbl2_372943966. Accessed: May 8, 2026.
- [16] F. Demir, D. A. Abdullah and A. Sengur, "A New Deep CNN Model for Environmental Sound Classification," in *IEEE Access*, vol. 8, pp. 66529-66537, 2020, doi: 10.1109/ACCESS.2020.2984903.
<https://doi-org.focus.lib.kth.se/10.1109/ACCESS.2020.2984903>

