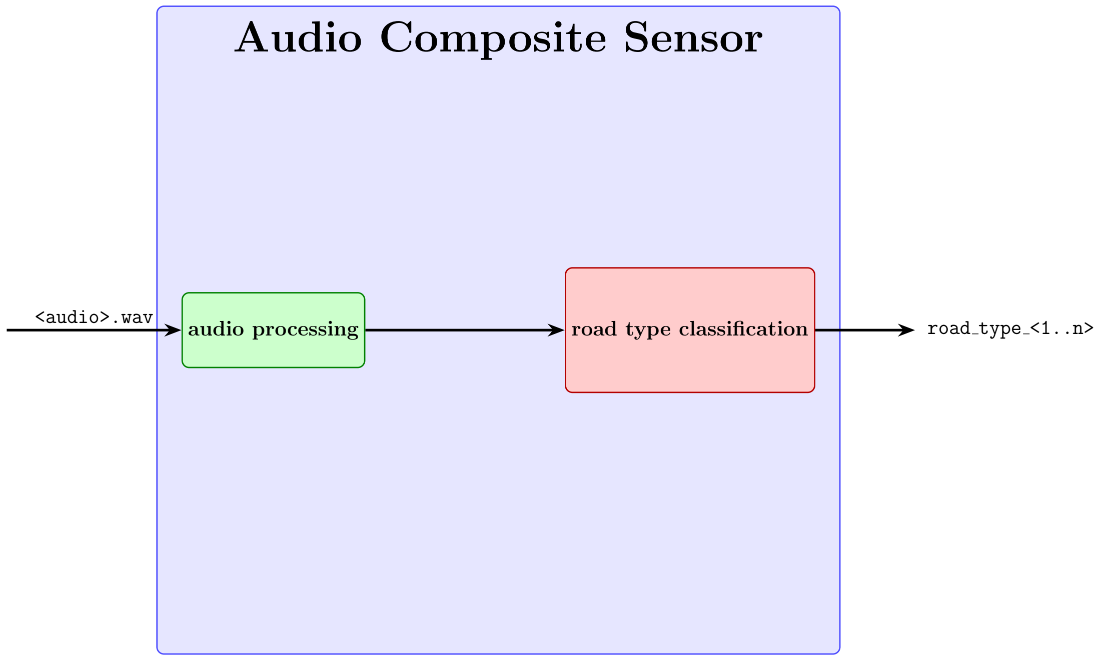


Audio Based Road Type Classification Using CNNs and AST

Development of Audio Based Road Type Classification Models with Focus on Convolutional Neural Networks and The Audio Spectrogram Transformer Model

Degree project report in Computer Engineering

Faisal Kohestani & Niloofar Mehrzad

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Degree project report 2025

Audio Based Road Type Classification Using CNNs and AST

Development of Audio Based Road Type Classification Models with Focus on Convolutional Neural Networks and The Audio Spectrogram Transformer Model

Faisal Kohestani & Niloofar Mehrzad

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden 2025

Audio Based Road Type Classification Using CNNs and AST
Development of Audio Based Road Type Classification Models with Focus on Convolutional Neural Networks and The Audio Spectrogram Transformer Model
Faisal Kohestani & Niloofar Mehrzad

Supervisor: Arsham Gholamzadeh Khoee, Computer Science and Engineering
Examiner: Nicholas Smallbone, Computer Science and Engineering

Degree project report 2025
Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Cover: A simple model of the audio composite sensor developed in this project it comprises of the following parts; input, audio processing, classification, and output.

Typeset in LaTeX
Gothenburg, Sweden 2025

Audio Based Road Type Classification Using CNNs and AST
Development of Audio Based Road Type Classification Models with Focus on Convolutional Neural Networks and The Audio Spectrogram Transformer Model
Faisal Kohestani & Niloofar Mehrzad
Department of Computer Science and Engineering
Chalmers University of Technology

Abstract

This thesis investigates the use of machine learning models for classifying road types based on vehicle audio recordings. The goal is to evaluate the effectiveness of different model architectures, specifically Convolutional Neural Networks (CNNs) and the transformer-based Audio Spectrogram Transformer model, in distinguishing between road surface types such as smooth asphalt, rough asphalt and uneven surfaces. Audio data was pre-processed using feature extraction techniques such as Mel-spectrograms and Mel-frequency cepstral coefficients (MFCCs). Multiple CNN models were developed and trained, while a pre-trained Audio Spectrogram Transformer model was fine-tuned for the task. All models were evaluated using stratified 5-fold cross-validation with performance measured through metrics such as accuracy, F1-score, precision, recall, confusion matrices and inference metrics. The results show that the AST model achieved the highest classification performance, while the CNN models offered advantages in inference speed and memory usage. Post-training quantization was applied to all models using Qualcomm's AI Hub to determine their viability for deployment on mobile or embedded-systems. The findings highlight the potential of this audio-based road type classification as a composite sensor for automotive applications. Limitations related to dataset, feature representation, and recording conditions are discussed, along with recommendations for future improvements and deployment strategies.

Keywords: Classification, CNN, AST, Audio, Mel-spectrogram, MFCC, Cross-validation, Road-type, Quantization.

Acknowledgements

We want to thank our supervisors at both Chalmers and Volvo Cars for helping us along the way and making this thesis report possible. We have learned very much by working with the people in Volvo Cars and in the research for this project. We also want to thank Chalmers University of Technology for giving us the opportunity to apply ourselves in projects outside of school and enable practical experiences for its students.

Faisal Kohestani & Niloofar Mehrzad, Gothenburg, June 2025

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AST	Audio Spectrogram Transformer
CNN	Convolutional Neural Network
MFCC	Mel-Frequency Cepstral Coefficient
ML	Machine Learning
NPU	Neural Processing Unit
FFT	Fast Fourier Transform

Nomenclature

Below is the nomenclature of parameters that have been used throughout this thesis.

Parameters

sr (Sampling rate)	The number of audio samples per second. It defines how finely audio is digitally represented
n_fft (FFT window size)	The size of the Fast Fourier Transform window, determining frequency resolution.
hop_length	The number of audio samples shifted forward between successive FFT windows.
n_mels	The number of Mel frequency bands used to represent frequencies.

Contents

List of Figures

List of Tables

1 Introduction

Modern vehicles increasingly rely on intelligent systems to enhance safety and comfort. One important aspect of this development is the ability to detect and classify road surface conditions, which can inform vehicle behavior and driver assistance systems. While many solutions rely on visual or physical sensors, this project explores an alternative approach, using audio signals captured from the vehicle to classify road types. The following sections present the background, purpose, goals, and limitations of this study, followed by the technical background, implementation, evaluation, and results.

1.1 Background

Volvo Cars is a global automotive manufacturer known for its emphasis on safety [1], innovation, and sustainability. Within Volvo Cars, the Safety Center is dedicated to advancing vehicle safety through research and development. This department plays a crucial role in analyzing driving conditions, testing safety measures, and implementing new technologies to enhance vehicle performance, vehicle safety, and safe traffic systems [2].

No person should come to harm driving any car. By drawing from a wider range of information sources this can enhance the car's ability to detect anomalies and support Volvo's commitment to protecting what's important. By understanding anomalies in the traffic infrastructure, such as slippery roads, recent incidents, or poor road maintenance, the car can better anticipate unexpected events. This may enable important, real-time insights that can inform not only the vehicle but also the traffic system as a whole.

1.2 Purpose

The purpose of this project is to define a composite sensor and provide a reliable method for classifying road types based on audio signals. Currently, road classification primarily relies on visual or sensor-based methods, such as cameras and accelerometers. However, these traditional approaches can be affected by factors such as poor lighting, obstructions, or the need for expensive specialized hardware.

Audio-based classification offers an approach that could complement existing techniques. Different road surfaces produce distinct acoustic signatures due to variations in texture, material, and condition. By analyzing these sound patterns generated by vehicle-road interactions, this method could serve as a cost-effective, possibly reliable and complementary solution to enhance road-type detection, particularly in scenarios where visual or sensor-based data is unreliable or impractical.

1.3 Goals

The main goals of this project are:

- Develop a machine learning model capable of classifying road types based on vehicle audio recordings. Also, to compare and evaluate different Machine Learning (ML) techniques, such as Convolutional Neural Networks (CNNs) and Audio Spectrogram Transformer (AST), to determine their effectiveness in audio-based road classification.
- Conduct an in-depth examination, assessment, and discussion of existing tool flows for deploying a pre-trained model on modern Automotive SoC (System On Chip).
- Compare the performance attainable on Neural Processing Unit (NPU) using a combination of pre-trained AST and CNN models.
- Define a composite sensor and provide recommendations on how this classification model could be used for applications in noise reduction strategies and vehicle safety.

Through these goals, we aim to deliver a well-tested and validated model that can serve as a foundation for further advancements in vehicle noise reduction, safety, and road classification technologies.

1.4 Limitations / Demarcations

To clearly define the scope and ensure a focused approach to achieving the project goals, several limitations have been established. First, the dataset used for this project is finalized, meaning no further audio recordings will be available during the duration of the project. This fixed dataset consists of a limited number of audio samples, 102 recordings, which leads to challenges such as data imbalance and constraints in dataset partitioning. Due to the small dataset size, it is only possible to create a training set and validation set, leaving insufficient data for a dedicated test set. Consequently, this limitation restricts the robustness of misclassification data analysis.

Additionally, the project explicitly concentrates on the CNN and AST models, excluding exploration of alternative machine learning techniques. This focus enables deeper analysis and optimization of selected models. Furthermore, all model development will strictly utilize Python, with small exceptions for certain audio analyses that require specialized software. These clearly defined limitations guide the project's scope, making it possible to achieve the target goals.

2
Technical Background

This chapter provides an overview of the key technical concepts and tools used in the project. The sections cover the primary Python libraries used to implement machine learning workflows, methodologies for audio data processing and feature extraction, theoretical foundations and architectures of the applied machine learning models, and finally, the evaluation metrics and techniques used to measure the models' performance. Each subsection introduces concepts essential for understanding the methods and results presented in later chapters.

2.1 Composite Sensors

Composite sensors are software-based sensors that process data collected from at least one (usually multiple) hardware sensors [3]. As seen in figure 2.1, in this project, the term composite sensor refers to the audio processing and machine learning model developed to classify road types based on audio signals generated by interactions between the vehicle and different road surfaces. Currently, the focus is solely on audio data, meaning a single hardware sensor is used. However, in the future, data from additional sensors like accelerometers could also be integrated.

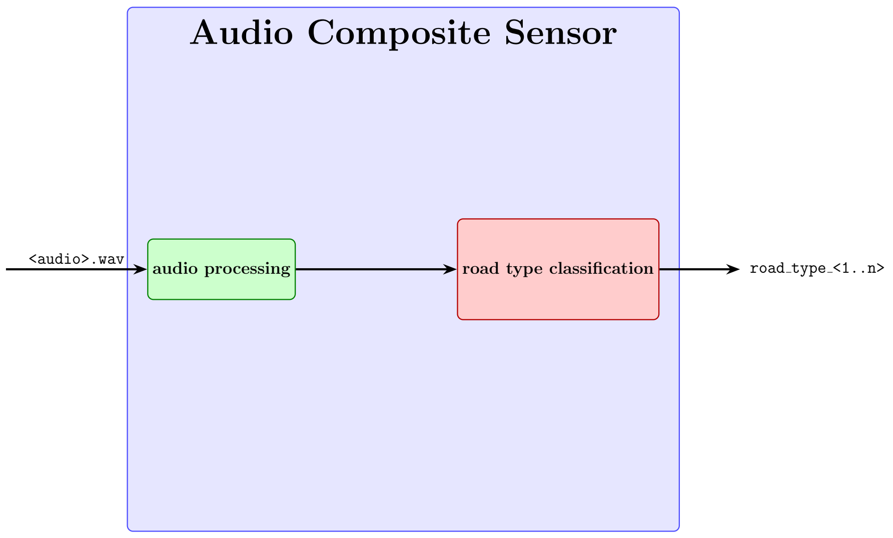


Figure 2.1: Simple overview of a hypothetical composite sensor for road type classification.

2.2 Python Libraries

Python is a high-level, object-oriented programming language renowned for data analysis and machine learning development. The subsections below detail the relevant libraries used for this study.

2.2.1 PyTorch and TorchAudio

PyTorch is an open-source deep learning framework widely used for developing and training deep neural networks. It is known for its flexibility, efficiency, and dynamic computational graph structure, making it easier to quickly develop and debug models [6]. PyTorch was chosen for this study due to its user-friendly interface, extensive documentation, and suitability for developing audio classification models.

TorchAudio is a powerful PyTorch library designed for audio and signal processing, offering essential tools for loading, transforming, and analyzing audio data [7]. It provides efficient I/O operations, signal processing functions, datasets, and pre-built models, making it ideal for machine learning applications. TorchAudio supports various audio formats and offers built-in functions for computing audio features such as Mel spectrograms, Mel-frequency cepstral coefficients (MFCCs), and other critical time-frequency transformations. Its direct compatibility with PyTorch made TorchAudio a natural choice for processing and preparing audio data in this project's workflow, which focuses on road-type classification using audio data.

2.2.2 Scikit-learn

Scikit-learn is an open-source machine learning library for Python. It provides efficient and user-friendly implementations of a broad range of algorithms for supervised and unsupervised learning, as well as tools for model evaluation, hyperparameter tuning, and data preprocessing [8]. Scikit-learn is particularly valuable for its ease of integration into machine learning workflows. It offers functions for data splitting, cross-validation, and computation of evaluation metrics. In this project, scikit-learn was utilized primarily for stratified data splitting (via StratifiedKFold) and evaluating model performance through metrics such as classification reports and confusion matrices. Its straightforward API and extensive documentation made it a natural choice for supporting model evaluation.

2.3 Audio Data Processing and Feature Extraction

Digital audio data must be transformed into suitable numerical representations before use in machine learning models. This section outlines fundamental audio signal processing concepts and techniques for extracting meaningful features from raw waveforms. These include sampling principles, time-frequency transformations, and feature representations commonly used in audio classification tasks. The following sections provide background on essential concepts such as the Nyquist–Shannon sampling theorem, short-time Fourier transform (STFT), Mel-spectrograms, and Mel-frequency cepstral coefficients (MFCCs).

2.3.1 Nyquist–Shannon Sampling Theorem

The Nyquist theorem, or sampling theorem, states that an analog signal must be sampled at least twice as fast as its highest frequency. This is necessary to accurately capture and reproduce the original signal without losing information [9]. This principle is crucial for digital audio systems to prevent aliasing. Ensuring accurate sampling rates is important for effectively analyzing audio features using techniques like Mel-spectrograms, making this theorem essential when resampling audio recordings.

2.3.2 Short-Time Fourier Transform (STFT)

Audio signals can be analyzed using the Short-Time Fourier Transform (STFT), which reveals how a signal's frequency content evolves over time. It divides the audio signal into smaller windows, then applies the Fourier Transform to each window. While commonly used, STFT's fixed window size can limit its ability to capture rapid signal changes [10]. In this project, STFT is indirectly applied through the Mel-spectrogram transform function in TorchAudio, affecting spectrogram resolution and size by adjusting parameters such as sampling rate, n_mels, n_fft, and hop length. This process helps extract frequency patterns associated with different road types, though limitations in capturing rapid frequency changes remain.

2.3.3 Mel-spectrograms and MFCCs

Spectrograms illustrate frequency changes over time by segmenting audio and applying Fast Fourier Transform (FFT) [13]. Mel-spectrograms use the Mel scale to align frequency bins with human auditory perception, emphasizing lower frequencies [13]. This makes them useful for audio classification by capturing relevant sound features. Mel-frequency cepstral coefficients (MFCCs) offer a more compact representation, emphasizing the broader spectral shape rather than detailed band variations [17, 16].

2.4 Machine Learning Models

This project focuses on two primary architectures: CNNs and transformer-based models, notably the AST model. CNNs effectively handle spatial or temporal patterns, while transformer models adapted for audio tasks offer improved capabilities for complex audio patterns.

2.4.1 Neural Networks and Convolutional Neural Networks

Artificial neural networks are a method inspired by the structure and function of human neurons to train machines [18]. They consist of many nodes, often called artificial neurons, that are connected to each other. Similar to the human brain, during training, connections between nodes (called synapses) are adjusted. When a neural network produces a good result, the connections responsible for that output are strengthened, making the network better at similar tasks in the future. A neural network (NN) model consists of at least three types of layers: an input layer, one or more hidden layers, and an output layer. In classification tasks, the number of neurons in the output layer typically matches the number of classes. In a neural network, each neuron processes the input it receives by applying mathematical functions, then forwards the output to the neurons in the next layer. Through the training process, the network fine-tunes the strength of the connections, which are known as weights between neurons. By continuously updating these weights, neural networks can recognize complex patterns within large datasets. These aspects form the foundation of deep learning, which enables the development of powerful architectures such as Convolutional Neural Networks (CNNs) [19].

CNNs are a type of deep learning architecture specifically designed to process image-based data. They are trained using convolutional layers and kernel filters for optimization. These layers extract features such as sharpness, edges, and complex patterns from the data to further process and compute the necessary operations for the final predictions [27]. With convolutional, pooling, and fully connected layers, CNNs are effective at image recognition, object detection, and even audio analysis based on spectrograms. With the help of a Mel-spectrogram, the CNN is able to detect patterns across time intervals and frequency components in the audio [13].

2.4.2 Audio Spectrogram Transformer (AST)

The AST model is a transformer-based deep learning model specifically designed for analyzing and classifying audio signals. Instead of working directly with raw audio, the AST model internally divides the raw waveform into chunks and then converts them into Mel-spectrograms, similar to how the Vision Transformer (ViT) handles visual data [24]. The AST model has proven highly effective for recognizing complex audio patterns, such as speech and environmental sounds, making it suitable for this project [24].

2.5 Qualcomm AI Hub

Qualcomm AI Hub is a cloud-based platform designed to simplify the deployment of machine learning models on Qualcomm devices, such as automotive hardware. It provides tools for compiling, optimizing, running inference, and reducing model size through quantization [14, 15].

In this project, Qualcomm AI Hub was used to gather inference metrics, compile models, quantize them, and run them on specific Qualcomm automotive embedded hardware to examine the viability of model deployment in such systems.

2.6 Evaluation Metrics and Techniques

To evaluate model performance, several methods were applied. Confusion matrices and classification reports were used to measure the models' classification performance across all classes, while inference metrics such as minimum inference time and peak memory usage were analyzed to examine the potential deployment viability of the models.

2.6.1 Confusion Matrix

A confusion matrix is commonly used for evaluating the performance of classification models. It summarizes prediction results by displaying the counts of true positives, true negatives, false positives, and false negatives in a matrix format. By analyzing the confusion matrix, it becomes possible to identify which classes the model tends to confuse, offering insights beyond a simple accuracy score. In multi-class problems, confusion matrices expand to compare multiple classes simultaneously, making them especially useful for classification tasks such as the one in this project.

2.6.2 Classification Report

A classification report provides a detailed breakdown of a classification model's performance using key metrics: **precision**, **recall**, **F1-score**, and **accuracy** for each class.

- **Precision** measures the proportion of correctly predicted positive observations relative to the total predicted positives.
- **Recall** measures the proportion of correctly predicted positives relative to all actual positives.
- **F1-score** is the harmonic mean of precision and recall.
- **Accuracy** measures the overall proportion of correct predictions.

The classification report allows a more nuanced evaluation, especially when the class distributions are imbalanced, as it highlights model strengths and weaknesses for each class individually.

2.6.3 Inference Metrics

Inference metrics such as minimum inference time and peak memory usage are important when considering deploying a model in embedded systems. These metrics determine whether a model is suitable or even functional on resource-constrained systems. The first metric, minimum inference time, refers to the minimum duration required for the model or application to process input and produce its output [25]. In the context of this study, it refers to the time needed for the model to classify a one-second segment of road noise into one of three classes. The peak memory usage is represented as a range (A - B MB), indicating the maximum runtime memory footprint observed during inference [25].

By measuring these two metrics, one can determine if the model in question is suitable for deployment in mobile or embedded systems.

3
Methods

This study employed a computational experimental approach to develop and evaluate three CNN models, comparing them to each other and the transformer-based AST model for audio classification. The methods were designed to systematically pre-process audio data, extract relevant features in the form of mel-spectrograms, and then train deep learning architectures capable of identifying patterns in the audio signals. The following sections describe the procedures used for data collection, pre-processing, model design, training, and evaluation.

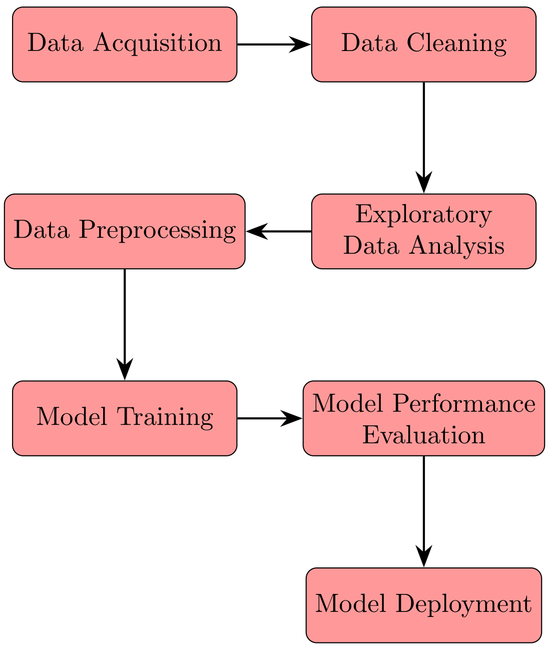


Figure 3.1: Overview of the core stages in development of the machine-learning models.

3.1 Tools, Technologies, and Workflow

Various tools and technologies were employed in this project, along with an agile workflow. Below is a summarized list and description of these:

- **Git and GitHub**: Git was used for version control, with a central repository hosted on GitHub, enabling team members to collaborate, edit, and review code effectively.

- **Audacity**: An open-source software for audio analysis, particularly useful for examining raw waveforms, listening to files, and performing frequency analyses on recordings.
- **Python and PyTorch**: Python served as the programming language, and PyTorch was chosen as the deep learning framework due to its powerful tools, including TorchAudio, which were essential for audio-specific processing tasks in this project.
- **Qualcomm AI Hub**: A cloud-based platform built on the Qualcomm AI Stack used to evaluate models' inference performance on embedded automotive systems.
- **Agile workflow**: Weekly meetings were held with Volvo and Chalmers supervisors to address technical questions, discuss progress, and set new sprint goals.

3.2 Dataset and Data Collection

The audio datasets were originally collected by the NVH team from Volvo Cars during road tests in Hällered, a dedicated Volvo testing area featuring various road surfaces. Although initially recorded for other research purposes, the data proved suitable for this project. Audio was recorded at a sampling rate of 48,000 Hz using the **Volvo XC40 EV** in three different ways:

- **Wheel-mic**: A microphone placed near the wheel, capturing high-quality road noise.
- **Engine-mic**: A microphone placed near the engine. Although this recording contained engine noise, it remained usable after pre-processing due to the electric car's quiet engine.
- **In-cabin-mics**: Four single-channel microphones inside the car capturing interior noise. These recordings were available only for certain road types but were used due to their clarity and absence of extraneous noise.

Recordings included six road types: *rough asphalt*, *smooth asphalt*, *cobblestone*, *Vienna pavement*, *Belgian pavement*, and *pipes*. As summarized in Table 3.1, the dataset was small and imbalanced, comprising 102 total recordings. Rough asphalt accounted for 69.6%, smooth asphalt for 21.6%, and other road types for 8.8%. Due to this imbalance, the classes were grouped as follows:

Class	# of files	Total length (s)	Avg. length (s)
Smooth asphalt	22	85	7.9
Rough asphalt	71	558	3.9
Others	9	48	5.3

Table 3.1: Overview of recording counts and duration for each road type class.

- **Rough asphalt**: Includes recordings on rough asphalt using all recording methods.
- **Smooth asphalt**: Includes recordings on smooth asphalt using all methods except in-cabin-mics.

- **Others**: Combines recordings of less common road types with similar sound profiles (cobblestone, Vienna pavement, Belgian pavement, pipes), excluding engine-mic recordings.

Due to limited data, only training (80%) and validation (20%) sets were created, without a separate test set.

3.3 Audio Analysis

Audio recordings were analyzed using *Audacity*, enabling inspection of waveforms and frequency analysis. It revealed high-energy frequencies predominantly ranged between 20–1400 Hz, influencing the decision to resample audio from 48 kHz to 3 kHz during CNN model pre-processing. This resampling reduced computational costs and aligned with the Nyquist–Shannon sampling theorem.

3.4 Pre-processing and Feature Extraction for CNN Models

CNN models typically require pre-processing and feature extraction using mel-spectrograms. The steps used were:

1. Load audio files using PyTorch.
2. Convert multi-channel audio to mono.
3. Resample audio from 48 kHz to 3 kHz.
4. Segment waveforms into 1-second chunks.
5. Optionally augment training set segments.
6. Transform waveforms into mel-spectrograms using TorchAudio's Mel-spectrogram function.

Augmentation factors varied by class: **smooth asphalt** (3x), **rough asphalt** (no augmentation due to it being the majority class by far), and **others** (7x). Augmentation methods included adding Gaussian noise (factor: 0.005), time stretching (rate: 0.8–1.2), pitch shifting (±3 semitones), and gain adjustments (±6 dB).

Mel-spectrogram parameters are detailed in Table 3.2, producing images of size 64×30:

Table 3.2: Mel-spectrogram Parameters

Parameter	**Value**
Sample rate	3,000 Hz
n_fft	512
Hop length	100
n_mels	64

3.5 Pre-processing and Feature Extraction for AST Model

The AST required minimal pre-processing due to its pretrained architecture. Audio was resampled from 48 kHz to 16 kHz, and the same augmentation options as for CNNs were applied.

3.6 Design of the Convolutional Neural Network Models

For this project, CNN architectures were chosen due to their proven effectiveness in capturing local patterns in spectrogram representations of audio signals. Three different CNN models were developed for the audio classification task: a basic custom CNN model, a custom residual neural network, and a pretrained CNN model (ResNet18). This section details the design of each CNN model.

3.6.1 Basic CNN Model

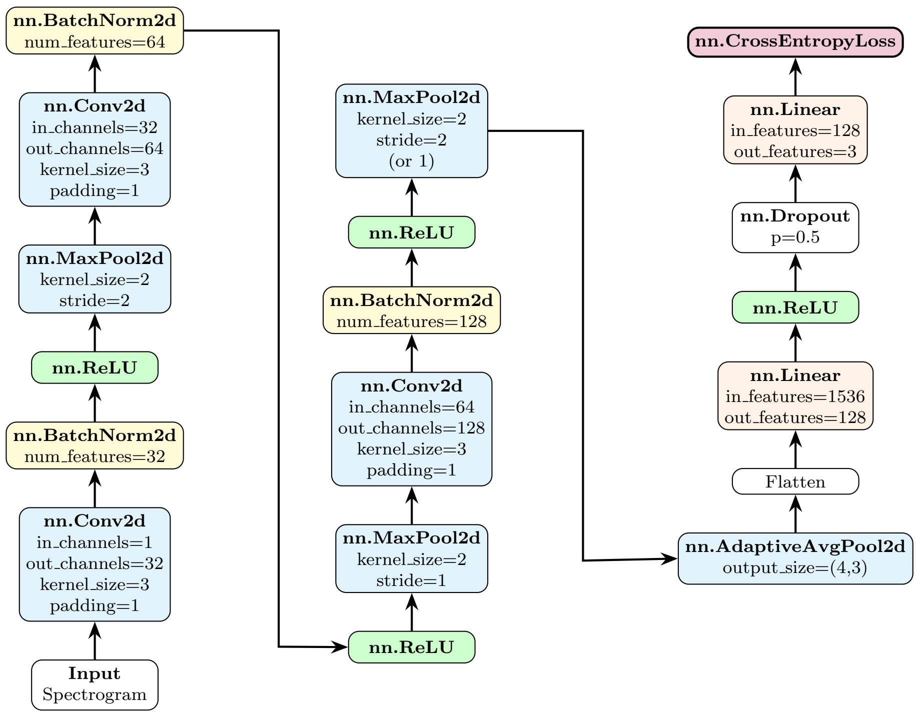


Figure 3.2: A diagram depicting the overall design of the Basic CNN model.

This simple CNN was designed specifically for the road type classification task. The network, as seen in figure 3.2, comprised three convolutional blocks, each including a 2D convolutional layer (kernel size 3x3, padding 1), followed by batch normalization and ReLU activation. Max pooling was applied after each block to reduce spatial dimensions, using an asymmetric kernel (2x1) in the second pooling layer to preserve frequency information. The third pooling layer dynamically adjusted its kernel size (2x2 or 2x1) based on the input size.

An adaptive average pooling layer compressed the feature map to a fixed output size (4x3) regardless of input dimensions. This was followed by a fully connected layer with 128 units and ReLU activation, a dropout layer with a rate of 0.5 to mitigate overfitting, and a final linear output layer.

3.6.2 Residual CNN Model

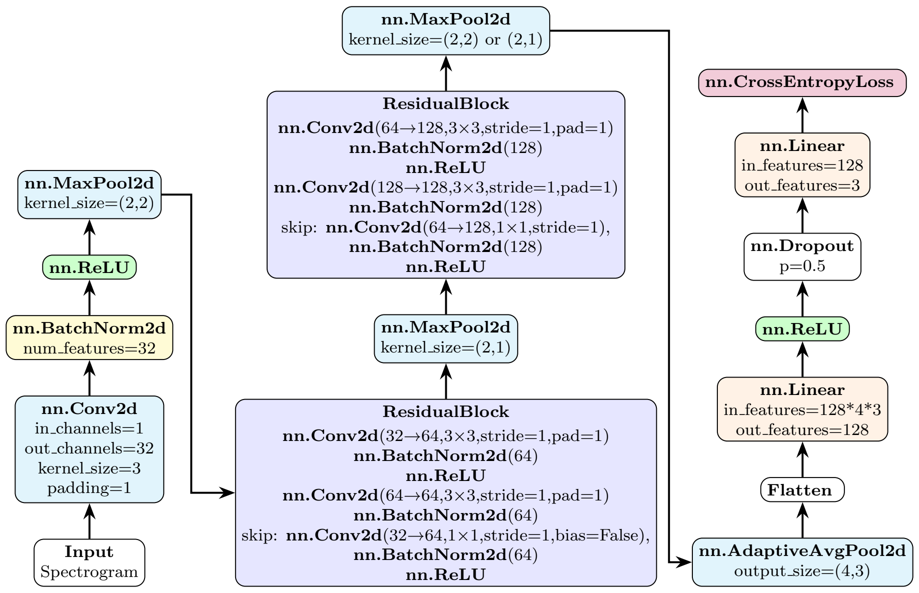


Figure 3.3: A diagram depicting the overall design of the Residual CNN model.

The residual CNN model, as seen in figure 3.3, incorporated residual connections to enhance feature learning and improve gradient flow. The model began with an initial convolutional layer (kernel size 3x3, 32 filters, padding 1), followed by batch normalization, ReLU activation, and max pooling. Two residual blocks followed, each containing two convolutional layers (3x3 kernel size, with batch normalization) and a skip connection. A 1x1 convolution was included in the skip connection when input and output dimensions differed.

Max pooling was applied after each residual block, with the final pooling layer dynamically selecting between a (2x2) or (2x1) kernel based on input spectrogram size. An adaptive average pooling layer standardized output dimensions to (4x3). The resulting feature maps were flattened and passed through a fully connected layer with 128 units, followed by a dropout layer with a rate of 0.5 and a final linear classification layer.

3.6.3 Pretrained CNN Model

To explore transfer learning benefits, a model based on the pretrained ResNet18 architecture was utilized. The original model, trained on ImageNet, was adapted by modifying the first convolutional layer to accept single-channel input for compatibility with grayscale spectrograms. The final fully connected layer was replaced to match the number of output classes.

Input spectrograms were resized to 224x224 pixels using bilinear interpolation, meeting ResNet's input dimension requirements. The pretrained ResNet architecture otherwise remained unchanged as a feature extractor.

3.7 Implementation of the Audio Spectrogram Transformer Model

To benchmark performance against convolutional architectures, Volvo Cars provided an Audio Spectrogram Transformer (AST) model. The AST is a transformer-based architecture pre-trained on the AudioSet dataset and fine-tuned for environmental sound classification. The pre-trained model "MIT/ast-finetuned-audioset-10-10-0.4593" was adapted specifically for the three-class classification task: **smooth asphalt**, **rough asphalt**, and **others**.

3.8 Training and Evaluation Setup

After model development, both CNN and AST models were trained and evaluated under standardized procedures to ensure fair comparisons. Conditions included cross-validation, data augmentation, and consistent evaluation metrics. Specific configurations and strategies used for CNN and AST models are detailed below.

3.8.1 CNN Models

Models were trained using supervised learning with stratified five-fold cross-validation, ensuring robust and generalizable outcomes given the limited and imbalanced dataset. The dataset was split into 80% training and 20% validation sets, stratified to preserve class distribution, with splits performed at the file level to prevent data leakage.

Spectrogram normalization was performed using the training set mean and standard deviation for each fold. Data augmentation was applied exclusively to the training set. Training lasted 10 epochs using the Adam optimizer with a learning rate of 0.001 and weight decay of 0.0001 to minimize overfitting. A step-based learning rate scheduler (StepLR) reduced the learning rate by half every 5 epochs. Training was executed in mini-batches of size 8, employing cross-entropy loss as the loss function. Each epoch tracked training accuracy, training loss, validation accuracy, and validation loss. Model performance was evaluated via confusion matrices and classification reports.

All models were implemented in PyTorch and trained on an Intel Core i7-10610U CPU @ 1.80GHz $\times$ 8, with checkpoints saved after each fold.

3.8.2 AST Model

Minimal changes were made to the provided AST code, specifically adding stratified cross-validation and training set augmentation, matching CNN methodologies. The dataset was encapsulated in a custom RoadNoiseDataset object, compatible with Hugging Face's **Trainer** API.

The AST model was fine-tuned for 5 epochs using the Hugging Face **Trainer** API. Training parameters included a batch size of 4, a learning rate of 2×10^{-5}, weight decay of 0.01, and a linear scheduler with 50 warmup steps. Evaluation and checkpoint saving occurred at each epoch's end, with the best model determined by validation accuracy. The highest-performing model across folds, along with label mapping for inference, was saved.

Model performance was evaluated for each fold using classification reports (precision, recall, F1-score, accuracy) and confusion matrices.

3.9 Quantization and Inference Using Qualcomm's AI Hub

To optimize inference time and memory efficiency, post-training quantization was performed via Qualcomm's AI Hub API, following the general workflow as seen in 3.4.

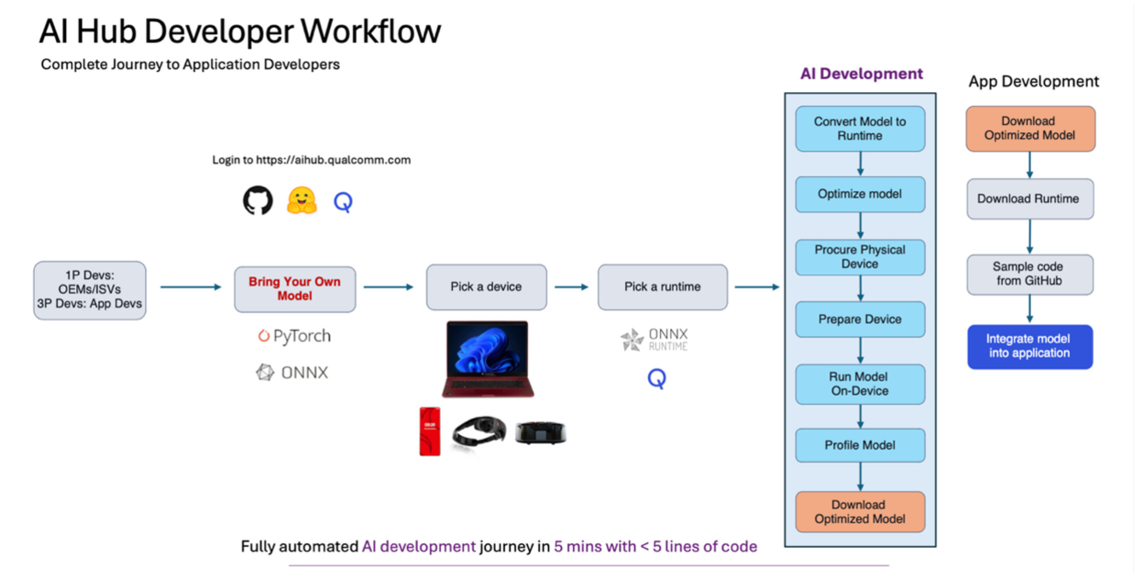


Figure 3.4: An overview of the AI-development workflow in Qualcomm AI Hub [26]. Reprinted with permission

Each model was initially traced using TorchScript with dummy input tensors matching expected input shapes (CNN models) or full audio file shapes (AST model). These traced models were compiled into ONNX format via Qualcomm AI Hub, targeting the **SA8255** proxy hardware platform. Calibration data from a representative dataset subset facilitated accurate quantization.

Quantization applied **INT8 precision** to both weights and activations, automatically handled by Qualcomm AI Hub's pipeline using the provided calibration data.

Qualcomm AI Hub API executed inference jobs on single audio files using the SA8255 Qualcomm device, collecting inference metrics such as minimum inference time and peak memory usage. Metrics were compared between quantized and non-quantized model versions.

4 Results

This section presents the outcomes of the experiments conducted with the CNN models and the AST model. The results focus on the models' learning behavior during training, their classification performance across folds, confusion matrices for visualizing classification performance, and inference-related metrics.

4.1 Classification Reports (Per-fold Median + Std)

This subsection summarizes the classification performance across all cross-validation folds. For each model, the median and standard deviation of key metrics—precision, recall, F1-score, and accuracy are reported to provide an overview of performance consistency across different data splits.

Table 4.1: Median and standard deviation of classification metrics across five cross-validation folds for each model

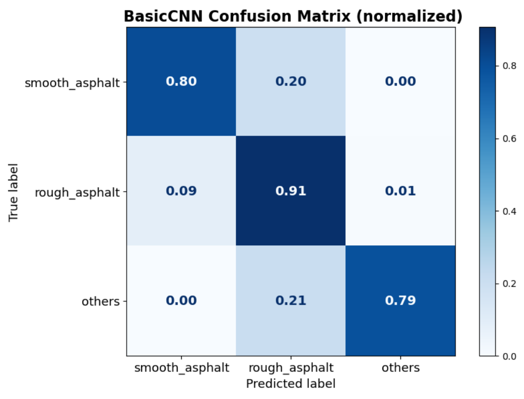
Model	Metric	Precision	Recall	F1-score	Support
Basic CNN	others	1.00 ± 0.20	1.00 ± 0.24	0.77 ± 0.16	11.0 ± 5.0
	rough_asphalt	0.97 ± 0.05	0.89 ± 0.07	0.93 ± 0.03	129.0 ± 9.5
	smooth_asphalt	0.68 ± 0.26	0.79 ± 0.23	0.75 ± 0.18	20.0 ± 7.4
	Accuracy		0.89 ± 0.05		160.0 ± 7.4
Residual CNN	others	1.00 ± 0.13	0.92 ± 0.26	0.83 ± 0.17	11.0 ± 5.0
	rough_asphalt	0.95 ± 0.06	**0.94 ± 0.08**	0.94 ± 0.04	129.0 ± 9.5
	smooth_asphalt	0.79 ± 0.29	0.74 ± 0.29	0.68 ± 0.26	20.0 ± 7.4
	Accuracy		0.90 ± 0.06		160.0 ± 7.4
Pretrained CNN	others	0.92 ± 0.27	0.91 ± 0.34	0.67 ± 0.26	11.0 ± 5.0
	rough_asphalt	0.96 ± 0.083	0.93 ± 0.09	0.86 ± 0.04	129.0 ± 9.5
	smooth_asphalt	0.53 ± 0.18	0.73 ± 0.29	0.38 ± 0.21	20.0 ± 7.4
	Accuracy		0.81 ± 0.06		160.0 ± 7.4
AST	others	**1.00 ± 0.00**	**1.00 ± 0.22**	**1.00 ± 0.15**	2.0 ± 0.55
	rough_asphalt	**1.00 ± 0.08**	0.93 ± 0.06	**0.96 ± 0.05**	14.0 ± 0.45
	smooth_asphalt	**0.80 ± 0.14**	**1.00 ± 0.22**	**0.89 ± 0.13**	4.0 ± 0.45
	Accuracy		**0.95 ± 0.06**		20.0 ± 0.55

As shown in Table 4.1, the AST model achieved the highest median F1-score and accuracy among all models. Among the CNN architectures, the Basic CNN and Residual CNN models obtained similar results across all classes. The Pretrained CNN performed the worst among the CNN models, with lower F1-scores for both

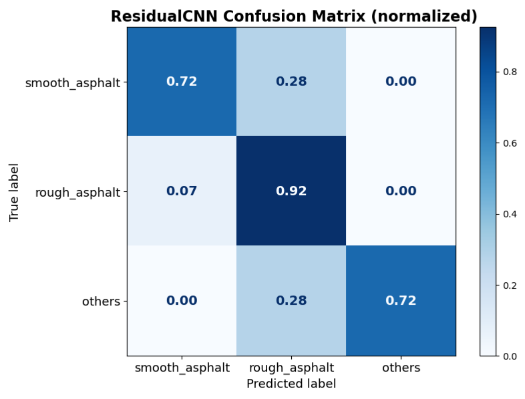
smooth asphalt and **others**, as well as lower overall accuracy. All models showed moderate to high variability in the minority classes **smooth asphalt** and **others**, while exhibiting low levels of variability across folds for the majority class **rough asphalt**.

4.2 Normalized Confusion Matrices

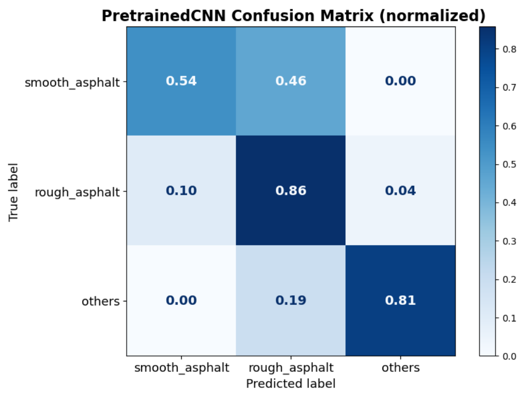
This subsection presents the normalized confusion matrices for each model, combining predictions across all validation folds. These matrices offer an overview of how well each model distinguished between target classes.



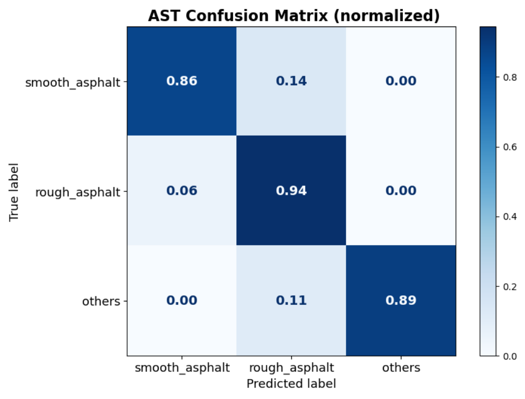
(a) Basic CNN model



(b) Residual CNN model



(c) Pretrained CNN model



(d) AST model

Figure 4.1: Normalized confusion matrices for all models, combining predictions across all validation folds

In the Basic CNN matrix in Figure 4.1a, many samples were correctly classified, although a moderate amount of confusion occurred between **rough asphalt** and **smooth asphalt**, as well as between **rough asphalt** and **others**—approximately 20% in both cases.

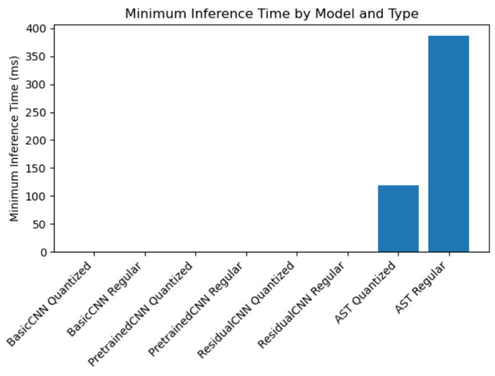
The Residual CNN matrix in Figure 4.1b shows similar results to the Basic CNN matrix for **rough asphalt**, but more confusion between **rough asphalt** and

smooth asphalt, as well as between **rough asphalt** and **others**, at 28% in both cases.

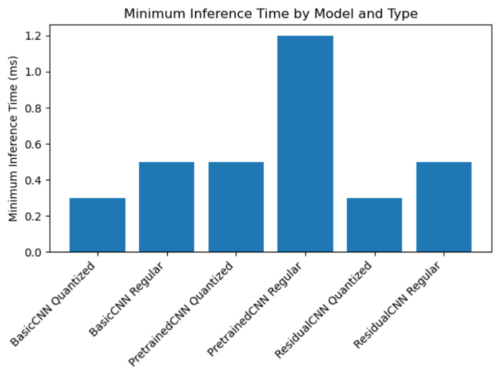
The Pretrained CNN model demonstrates slightly weaker results for the **rough asphalt** class, slightly better class separation for the **others** class, but the most confusion, among all figures in 4.1, for the **smooth asphalt** class. It misclassified **smooth asphalt** samples as **rough asphalt** 46% of the time.

4.3 Inference Metrics

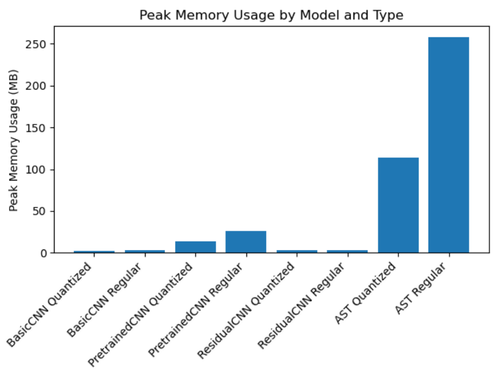
This subsection reports the inference-related metrics when running the models on the Qualcomm SA8225 (proxy) device. The metrics include peak memory usage and expected minimum inference time. These results provide insight into the practical deployment aspects of the models beyond their predictive performance.



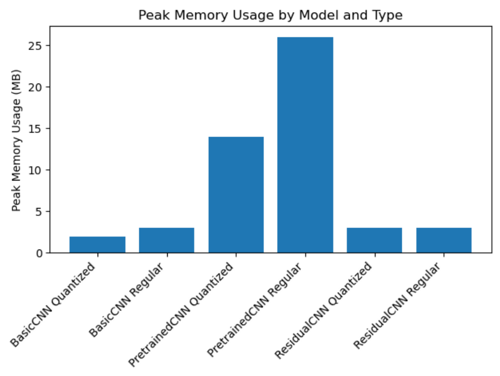
(a) Minimum inference time for all models



(b) Minimum inference time for all models, excluding the AST model



(c) Peak memory usage for all models



(d) Peak memory usage for all models, excluding the AST model

Figure 4.2: Comparisons of all models (both regular and quantized versions) and their minimum inference time in ms, and peak memory usage in MB, running on the Qualcomm SA8225 (proxy) device.

To highlight differences clearly, two versions of each bar chart were produced: one including all models (Basic CNN, Residual CNN, Pretrained CNN, and AST), and one excluding the AST model to better visualize the differences among the CNN architectures.

Figures 4.2a and 4.2b display the minimum inference time across models, while Figures 4.2c and 4.2d in Figure 4.2 present the peak memory usage. When including the AST model, its inference time and memory usage are significantly higher compared to the CNN models, overshadowing their differences. In the charts excluding the AST model, it can be seen that the Basic CNN model achieves the lowest minimum inference time and estimated peak memory usage. The Residual CNN shows nearly identical results, with only slightly higher peak memory usage in its quantized version. The Pretrained CNN model has the highest resource demands among the CNN models, while the AST model has the highest resource demands, by a large margin, compared to all of the other models.

5 Discussion

This section discusses and interprets the results presented in the previous chapter. The analysis focuses on model performance, data quality, feature representation, and the practical implications of deploying the models in a mobile environment. Additionally, the limitations of the models and dataset are examined, and recommendations for future work are proposed.

5.1 Project Purpose and Broader Context

The primary goal of this study was to explore and develop machine learning models, specifically CNNs and the AST model, for road type classification. The purpose was to potentially improve noise reduction strategies in vehicles as well as enhance road awareness and safety.

From a practical perspective, using sound as a sensor is reasonable since it is low-cost, non-intrusive, and simple to implement in real-time systems, including real-time data collection. Compared to vision-based sensors or systems, audio sensors are less affected by visual obstructions such as poor lighting or dirt. However, audio sensors have their own challenges, including sensitivity to background noise, variation between vehicle types, and the need for robust signal processing.

Overall, this project contributes to the ongoing development of intelligent sensing systems by demonstrating how audio signals can be processed and classified using modern machine learning techniques. The findings and insights gained here can serve as a foundation for future work aiming to implement real-time, audio-based road type classification systems in real-world vehicles.

5.2 Data Processing and Feature Extraction

This section discusses how audio data was processed for training and evaluating the models, as well as the relative effectiveness of different feature representations, such as Mel-spectrograms and MFCCs.

The audio data pre-processing procedures differed between the two model architectures: AST and CNN. The AST model required minimal pre-processing due to its built-in data processing capability, enabling it to internally segment audio files into chunks and convert them to Mel-spectrograms before training [24]. The necessary pre-processing steps were converting audio into mono-sound and resampling it from 48 kHz to 16 kHz.

The CNN models required additional pre-processing steps, as they lacked internal audio processing capabilities. The pre-processing pipeline involved converting audio to mono, resampling from 48 kHz to 3 kHz, segmenting audio into 1-second chunks, and transforming them into Mel-spectrograms. The 3 kHz resampling rate was chosen based on the frequency range (20–1400 Hz) of the relevant audio signals, adhering to Nyquist's theorem [36]. The 1-second segment length was selected based on practical considerations for real-time applications and dataset size limitations.

Mel-spectrograms were chosen as the primary feature representation because of their established effectiveness in audio classification tasks, particularly for CNN models [30]. Mel-spectrograms capture both temporal and spectral patterns, making them suitable for CNN-based classification tasks.

Experiments with MFCC feature representations were also conducted to assess their performance. Although MFCCs require less computation and produce more compact representations [31], they lose significant 2D structural information that could negatively affect classification performance, especially when distinguishing classes with similar audio profiles. The MFCC-based model demonstrated poorer performance, particularly in distinguishing between rough and smooth asphalt, validating the decision to prioritize Mel-spectrograms.

5.3 Model Analysis and Comparison

This section compares the performance and behavior of the developed models, considering metrics such as overall accuracy, per-class metrics, and performance stability across cross-validation folds.

Given the dataset imbalance, overall accuracy alone was not sufficient for model comparison. Instead, both overall accuracy and per-class metrics such as F1-score were used. Table 4.1 shows that the AST model outperformed the CNN models in almost every metric, supported by confusion matrix 4.1d in figure 4.1. The AST model's superior performance was expected due to its transformer-based architecture specifically optimized for audio tasks. However, its complexity and resource intensity present practical challenges for deployment in mobile environments, an issue discussed further in the following subsection.

Among the CNN models, the Basic CNN and Residual CNN performed similarly, while the pretrained ResNet18 model performed notably worse. The Basic and Residual CNN models showed comparable accuracy and stability, likely due to their architectural similarities. A notable limitation in the Residual CNN model was that its depth was not fully exploited to enhance feature extraction capabilities. Future work could improve this by designing deeper, more balanced residual architectures to leverage their advantages fully without compromising mobile compatibility.

In detailed confusion matrix analyses (figure 4.1), the Basic CNN slightly outperformed the Residual CNN, particularly in distinguishing smooth asphalt from rough asphalt. Conversely, the pretrained ResNet18 model showed significant performance degradation, especially for smooth asphalt classification, likely due to its generic image classification design and the resizing-induced loss of detail from spectrogram images.

All models consistently performed best on the rough asphalt class, likely due to data abundance and the limitations of the cross-entropy loss function used, which can be biased toward majority classes [33]. Future implementations could benefit from alternative loss functions like focal loss [34], which specifically addresses class imbalance by emphasizing hard-to-classify samples.

5.4 Inference and Deployment Considerations

This section evaluates the practical deployment aspects of the models, focusing on inference speed, memory usage, and suitability for mobile or embedded environments.

As shown in figure 4.2, the Basic CNN model exhibited the lowest minimum inference time and peak memory usage on the Qualcomm SA8225 (proxy) device, both in its regular and quantized forms. The Residual CNN model demonstrated similar performance, with only a slight increase of 1 MB in peak memory usage for its quantized version.

Among the CNN models, the Pretrained ResNet18 model required significantly more resources, approximately double compared to the other CNN models, due to its larger and more complex architecture.

The AST model was the most resource-intensive of all, in both regular and quantized versions. As observed in figure 4.2, its minimum inference time (387 ms) and peak memory usage (258 MB) greatly exceeded those of the CNN models. Even the quantized version had relatively high resource demands, with a minimum inference time of 120 ms and peak memory usage of 114 MB. The substantial resource requirement was expected, given its larger, transformer-based architecture and the lack of aggressive down-sampling beyond the initial 16 kHz resampling during preprocessing.

Given these considerations, the AST model is unsuitable for mobile or embedded systems due to its high computational demands. Conversely, the CNN models, particularly the Basic CNN model, are more appropriate for these environments due to their lower resource requirements. Future work could involve further optimization and improvement of CNN models, especially with additional high-quality audio data.

5.5 Dataset Quality, Improvements, and Limitations

This section examines challenges and limitations associated with the dataset, including class imbalance and recording setups, as well as their impact on model performance.

The dataset used in this study had two significant issues: a small overall size and notable class imbalances. Out of the 102 audio files, 71 were **rough asphalt**, 22 were **smooth asphalt**, and only 9 belonged to the **others** class. Additionally, there was an imbalance in audio file lengths, with an average length of approximately 8 seconds for **rough asphalt**, 4 seconds for **smooth asphalt**, and 5 seconds for **others**. These imbalances resulted in difficulties in developing models that performed consistently well across all classes, reflected in higher variability in classification metrics for minority classes, as seen in table 4.1 and figure 4.1.

The small dataset size also increased the risk of overfitting. To counteract this, data augmentation methods such as pitch shifting, time stretching, gain augmentation, and adding Gaussian noise were applied, primarily targeting minority classes. However, excessive augmentation risked introducing unrealistic data, potentially worsening model performance and overfitting, as highlighted by prior studies [35].

Due to these dataset limitations, a stratified five-fold cross-validation approach was employed instead of a simple test-train split. This method provided more robust and insightful results by ensuring balanced class distributions and enabling the observation of performance variability across different data splits.

6 Conclusion

This project set out to explore the use of machine learning techniques, specifically CNNs and the transformer-based AST model, for classifying road types using vehicle audio recordings. Through a series of experiments, models were developed, trained, and evaluated using stratified 5-fold cross-validation to ensure more robust performance estimation. Feature extraction techniques, such as Mel-spectrograms and MFCCs, were applied to transform raw audio data into structured inputs suitable for model training.

Results showed that all models achieved moderate performance, with the AST model outperforming the CNN architectures in both classification accuracy and consistency. However, these results were not entirely reliable due to limitations in the available dataset, which provided limited training data and a small validation set. Aside from classification performance, the CNN models offered advantages in terms of inference speed and memory usage, making them more suitable for real-time deployment on embedded or mobile systems. The quantization of models further supported the feasibility of deploying such systems in embedded environments, including Qualcomm-based devices used in vehicles.

In addition to technical performance, the project highlighted the importance of high-quality and extensive datasets, appropriate data augmentation, and effective feature representations. It also raised broader considerations about real-world deployment, such as recording conditions and inference efficiency.

Overall, this project demonstrates the potential of audio-based road type classification and contributes both practical results and methodological insights. With continued improvements in data collection, feature engineering, and model optimization, such systems could play an important role in enhancing vehicle awareness, comfort, and safety in the future.

6.1 Future Work

While this project demonstrated the viability of machine learning models for audio-based road type classification, several opportunities exist for further improvement.

A critical future direction is dataset enhancement. The dataset could benefit from increased volume, improved class balance, and extended audio durations. A comparative example is the dataset used in a research paper by another automotive company [37], containing 18,350 audio recordings, significantly larger than the dataset in this study. Increasing dataset diversity in road types, weather conditions, tire types, and vehicle models would also improve model robustness. Adding contextual metadata, such as road surface indices and tire pressure, and using a consistent audio recording setup (preferably the high-quality wheel-microphone setup) would further enhance data quality.

On the modeling side, exploring additional architectures, such as attention-based CNN hybrids or lightweight models tailored for embedded systems, could yield beneficial results.

Future research could also investigate alternative or hybrid feature extraction methods combining Mel-spectrograms and MFCCs, or other advanced time-frequency representations.

Finally, further steps towards real-world deployment should focus on optimizing inference speed, memory efficiency, and model quantization, preparing these systems for practical implementation within automotive infotainment environments.

Bibliography

[1] Volvo Cars Arrowhead, “Why Volvo is the Safest Car Brand,” 2025. [Online]. Available: `https://www.volvocarsarrowhead.com/why-volvo-is-the-safest-car-brand.htm`. [Accessed: Apr. 07, 2025].

[2] Volvo Cars, “Two decades in the service of saving lives: Volvo Cars Safety Centre celebrates 20 years,” 2020. [Online]. Available: `https://www.media.volvocars.com/global/en-gb/media/pressreleases/275423/two-decades-in-the-service-of-saving-lives-volvo-cars-safety-centre-celebrates-20-years`. [Accessed: Apr. 07, 2025].

[3] Android Open Source Project, “Sensor types,” Android Open Source Project documentation. [Online]. Available: `https://source.android.com/docs/core/interaction/sensors/sensor-types`. Last updated: Apr. 04, 2025. [Accessed: May. 14, 2025].

[4] K. Lu, J. Li, X. An, and H. He, “Vision Sensor-Based Road Detection for Field Robot Navigation,” *Sensors*, vol. 15, no. 11, pp. 29594–29617, 2015, doi: 10.3390/s151129594.

[5] Python Software Foundation, “Python Introduction.” [Online]. Available: `https://www.python.org/doc/essays/blurb/`. [Accessed: Feb. 25, 2025].

[6] NVIDIA, “What is PyTorch? – NVIDIA Glossary,” 2025. [Online]. Available: `https://www.nvidia.com/en-us/glossary/pytorch/`. [Accessed: Feb. 25, 2025].

[7] PyTorch Team, “torchaudio: PyTorch library for audio processing,” 2024. [Online]. Available: `https://pytorch.org/audio/stable/index.html`. [Accessed: Feb. 25, 2025].

[8] Codecademy, “Scikit-learn,” 2024. [Online]. Available: `https://www.codecademy.com/article/scikit-learn`. [Accessed: Feb. 25, 2025].

[9] H. J. Landau, “Sampling, data transmission, and the Nyquist rate,” *Proc. IEEE*, vol. 55, no. 10, 1967. [Online]. Available: `https://ieeexplore.ieee.org/document/1447892`. [Accessed: Apr. 07, 2025].

[10] J. B. Allen, “Short Term Spectral Analysis, Synthesis, and Modification by Discrete Fourier Transform,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 25, no. 3, pp. 235–238, 1977, doi: 10.1109/TASSP.1977.1162950.

[11] ScienceDirect, “Short-Time Fourier Transform,” 2025. [Online]. Available: `https://www.sciencedirect.com/topics/engineering/short-time-fourier-transform`. [Accessed: Feb. 25, 2025].

[12] Z. Khodzhaev, “A Practical Guide to Spectrogram Analysis for Audio Signal Processing,” arXiv preprint arXiv:2403.09321, 2024. [Online]. Available: `https://arxiv.org/abs/2403.09321`.

[13] B. Zhang, J. Leitner, and S. Thornton, “Audio Recognition using Mel Spectrograms and Convolution Neural Networks,” Dept. Elect. & Comp. Eng., Univ. of California, San Diego, Tech. Rep., 2019. [Online]. Available: `https://noiselab.ucsd.edu/ECE228_2019/Reports/Report38.pdf`. [Accessed: Apr. 07, 2025].

[14] Qualcomm AI Hub, “Qualcomm® AI Hub", [Online]. Available: `https://app.aihub.qualcomm.com/docs/`. [Accessed: Apr. 20, 2025].

[15] Qualcomm AI Hub, “Quantization (Beta)". [Online]. Available: `https://app.aihub.qualcomm.com/docs/hub/quantize_examples.html`. [Accessed: Apr. 20, 2025].

[16] Z. K. Abdul and A. K. Al-Talabani, “Mel Frequency Cepstral Coefficient and its Applications: A Review,” *IEEE Access*, 2022. [Online]. Available: `https://ieeexplore.ieee.org/document/9955539`. [Accessed: Apr. 07, 2025].

[17] N. A. Meseguer, “Speech Analysis for Automatic Speech Recognition,” Master’s thesis, Dept. Electronics & Telecommunications, Norwegian University of Science and Technology, Jul. 2009. [Online]. Available: `https://www.researchgate.net/profile/Vladimir-Kulchitsky/post/Speech_recognition_How_to_analyze_vocal_responses_automatically/attachment/59d632c479197b8077990788/AS%3A371736848683010%401465640379360/download/FULLTEXT01.pdf`. [Supervisor: Torbjørn Svendsen, IET].

[18] L. Hardesty, “Explained: Neural networks,” MIT News Office, Apr. 14, 2017. [Online]. Available: `https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414`. [Archived from the original on 18 Mar. 2024; retrieved 2 June 2022].

[19] C. M. Bishop, *Pattern Recognition and Machine Learning.* New York: Springer, 2006.

[20] IBM, “Convolutional Neural Networks,” 2025. [Online]. Available: `https://www.ibm.com/think/topics/convolutional-neural-networks`. [Accessed: Feb. 25, 2025].

[21] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *J. Big Data*, vol. 8, no. 1, p. 53, 2021, doi: 10.1186/s40537-021-00444-8.

[22] K. O’Shea and R. Nash, “An Introduction to Convolutional Neural Networks,” arXiv preprint arXiv:1511.08458, 2015. [Online]. Available: `https://doi.org/10.1186/s40537-021-00444-8`. [Accessed: Apr. 07, 2025].

[23] B. Zhang, J. Leitner, and S. Thornton, “Audio Recognition using Mel Spectrograms and Convolution Neural Networks,” Univ. of California, San Diego, Tech. Rep., 2020. [Online]. Available: `https://noiselab.ucsd.edu/ECE228_2019/Reports/Report38.pdf`. [Accessed: Apr. 07, 2025].

[24] Y. Gong, Y.-A. Chung, and J. Glass, “AST: Audio spectrogram transformer,” arXiv preprint arXiv:2104.01778, 2021. [Online]. Available: `https://arxiv.org/abs/2104.01778`. [Accessed: Feb. 25, 2025].

[25] Qualcomm AI Hub, “How it works,” Qualcomm AI Hub documentation. [Online]. Available: `https://app.aihub.qualcomm.com/docs/hub/howitworks.html`. [Accessed: Apr. 20, 2025].

[26] What is the typical developer worflow when using AI Hub?. [Online]. Available: `https://app.aihub.qualcomm.com/docs/hub/faq.html#what-is-the-typical-developer-worflow-when-using-ai-hub`. [Accessed: May. 19, 2025].

[27] A. Lindholm, N. Wahlström, F. Lindsten, and T. B. Schön, *Machine Learning: A First Course for Engineers and Scientists*. Cambridge: Cambridge University Press, 2022. [Pre-publication version, July 8; free to view and download for personal use only.]

[28] GeeksforGeeks, "Confusion Matrix in Machine Learning," 2025. [Online]. Available: `https://www.geeksforgeeks.org/confusion-matrix-machine-learning/`. [Accessed: Feb. 25, 2025].

[29] IBM, "Overfitting vs. Underfitting," 2025. [Online]. Available: `https://www.ibm.com/think/topics/overfitting-vs-underfitting`. [Accessed: Feb. 25, 2025].

[30] A. Maccagno *et al.*, "A CNN approach for audio classification in construction sites," in *Progresses in Artificial Intelligence and Neural Systems*, A. Esposito *et al.*, Eds., Smart Innovation, Systems and Technologies, vol. 184. Singapore: Springer, 2021, pp. 371–381, doi: $10.1007/978-981-15-5093-5\_33$.

[31] P. Jain, A. Sar, T. Choudhury, V. Singh, and K. Kotecha, "Differentiation of Music Genre from an Audio File Using Neural Networks," in *AI Technologies for Information Systems and Management Science*, L. Garg *et al.*, Eds., Lecture Notes in Networks and Systems, vol. 1136. Cham: Springer, 2024, pp. 482–490, doi: $10.1007/978-3-031-70789-6\_40$.

[32] K. Zhang, Y. Guo, X. Wang, D. Chang, Z. Zhao, Z. Mazziotta, and T. X. Han, "Competing ratio loss for discriminative multi-class image classification," *Neurocomputing*, vol. 464, pp. 473–484, 2021, doi: 10.1016/j.neucom.2021.08.106.

[33] S. Rajaraman, G. Zamzmi, and S. K. Antani, "Novel loss functions for ensemble-based medical image classification," *PLOS ONE*, vol. 16, no. 12, Art. no. e0261307, 2021, doi: 10.1371/journal.pone.0261307.

[34] J. Dong, "Focal Loss Improves the Model Performance on Multi-Label Image Classifications with Imbalanced Data," in *Proc. 2nd Int. Conf. Industrial Control Network And System Engineering Research*, 2020, pp. 18–21, doi: 10.1145/3411016.3411020.

[35] A. Nshimiyimana, "Acoustic data augmentation for small passive acoustic monitoring datasets," *Multimedia Tools and Applications*, vol. 83, pp. 63397–63415, 2024, doi: 10.1007/s11042-023-17959-2.

[36] A. V. Oppenheim and R. W. Schafer, "Sampling of Continuous-Time Signals," in *Discrete-Time Signal Processing*, 3rd ed., ch. 4, pp. 105–162. Upper Saddle River, NJ, USA: Prentice Hall, 2009.

[37] D. Yang, D. Zhang, Y. Yuan, Z. Lei, B. Ding, and L. Bo, "Road terrain recognition based on tire noise for autonomous vehicle," *Scientific Reports*, vol. 14, no. 1, Art. no. 30913, 2024, doi: 10.1038/s41598-024-81666-7.