Journal of Theoretical and Applied Information Technology
15th December 2025. Vol.103. No.23
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AUTOMATED CLASSIFICATION AND COUNTING OF VEHICLES USING DEEP LEARNING APPROACH

AMIR RAZA[1], JOHARI ABDULLAH[2], REHMAN ULLAH KHAN[3*], IRWANDI HIPNI MOHD HIPINY[4], HAMIMAH UJIR[5]

[1,2,4,5,] Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Sarawak, Malaysia
[3]Faculty of Cognitive Sciences and Human Development, Universiti Malaysia Sarawak, Sarawak, Malaysia
E-mail: [1]razaamir788@gmail.com, [2]ajohari@unimas.my, [3*]rehmanphdar@gmail.com, [4]mhihipni@unimas.my, [5]uhamimah@unimas.my

ABSTRACT

The rapid increase in automobile traffic in and around cities presents significant challenges for transportation management, infrastructure planning, and overall system viability. Conventional vehicle classification and counting techniques, which are mostly manual or sensor-based, are often constrained by inefficiency, labor intensity, and limited scalability in real-time applications. This paper proposes a deep learning approach for real-time classification and enumeration of vehicles using one-dimensional signals from piezoelectric sensors. The 1D-CNN was trained, using convolutional layers with small kernel sizes to stack, so that the temporal dependencies of sensor signals could be well learned, pooling and fully connected layers used to extract features with high strength. The model was trained using a hybrid dataset with field-collected sensor data and synthetically generated signals generated with traffic simulation tools to cover a classes of vehicles and road conditions, which guarantees the ability to scale and generalization. The proposed model is shown to perform well as it has a classification accuracy of 0.99, and mean Average Precision (mAP) of 0.98 on five different vehicle classes. Moreover, its performance was checked with the unseen data, which proved high generalization ability. The solution has potential to be deployed on edge computing devices because it is computationally efficient to support a realistic application of traffic monitoring.

Keyword: *Vehicle Classification, Vehicle Counting, 1D Convolutional Neural Network (1D-CNN), Road Energy-Harvesting Sensors, Intelligent Transportation System (ITS)*

1. INTRODUCTION

Road transportation is considered as one of the key drivers in economic and social development in any city. It plays a vital role in world commerce where companies rely on road transport to deliver goods and individuals are enabled to trade the online orders. As a developing country, Malaysia considers the road transportation network vital for economic development. The total number of vehicles on the road is expected to increase in many cities across Malaysia. Rapid population growth, rising economic activity, and increasing employment rates have contributed to the surge in vehicle numbers. The convenience and flexibility of owning private vehicles have led to a surge in vehicle demand. According to the Malaysia Automotive Association (MAA), the total number of vehicles registrations in Malaysia hit 36.3 million units [1]. The growth rate of vehicles indicates that there is the need to properly monitor traffic flow to enable proper planning and management of the road in terms of expansion, rerouting, installation of traffic lights and so on.

Figure 1: Roadside Vehicle Census Using Human Personnel.

Journal of Theoretical and Applied Information Technology
15th December 2025. Vol.103. No.23
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The growing number of vehicles presents a critical challenge for city administration bodies and connected organizations such as city councils and transport experts. Efficient administration of road schemes demands accurate listening of traffic flow to inform decisions concerning infrastructure growth, traffic signal installations, and city preparation initiatives. Traditional vehicle monitoring methods, such as roadside manual tools, are labor-intensive, prone to delays, and susceptible to errors. Additionally, they entail functional risks in antagonistic weather conditions and acquire solid costs, limiting their regularity and opportunity.

Key applications of traffic info include traffic signal accumulation, traffic volume broadcasting, city planning, and property allocation for traffic administration all along peak hours. Essential to these requests is the capability to classify and count differing types of vehicles accurately. Various vehicle types, to a degree heavy trucks, cars, motorcycles, and service cars, provide important insights for foundation preparation, traffic management, and tangible impact amounts. For instance, roads usually secondhand by heavy trucks can require supported building due to larger fundamental demands, requiring evidence-located in charge supported by correct vehicle type info.

The functionality of road networks requires smooth stream of vehicles while planning plays a significant role in attaining sustainable development [2]. This section explains the issues related to the management of road transportation and presents a case for using conventional, mechanical, and artificial intelligence-based methods for identification of car types and counts. As shown in Figure 2, various methods for vehicle classification can be categorized into conventional, mechanical and digital/IT approaches namely manual survey, sensor-based method and AI based methods including CNN and RNN.

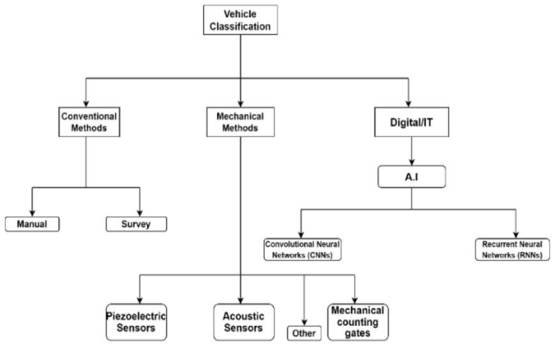

Figure 2: Vehicle Classification Taxonomy

Until now, vehicle classification and counting can be achieved through manual and survey methods which requires a lot of effort and time in data gathering [3]. These methods involve the use of operators to detect and segment out vehicles on images or frames extracted from videos and images. Surveys are used in gathering information on traffic distribution, types of vehicles and numbers [4]. The conventional methods are time taking, effortful, more of approximations and not very suitable for use when immediate results are required. Figure 3 shows the process of manual vehicle classification consisting of observation, human based analysis and generate data report for vehicle type and count.

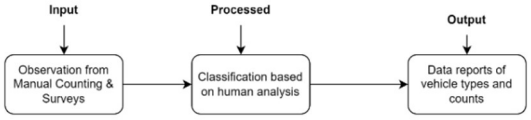

Figure 3: Conventional Methods Process Flow Diagram

To address these issues, the proposed study utilizes a deep learning-based vehicle classification and counting model that makes use of one-dimensional piezoelectric sensor signals. In particular, the research is informed by the following research questions:

1. Does a one-dimensional Convolutional Neural Network (1D-CNN) produce the desired classification of the different types of vehicles upon piezoelectric sensor signals?
2. Does hybrid dataset, which involves real world data and synthetically generated signals, enhance classification performance and robustness at different traffic conditions?

3. Does the proposed framework attain computational efficiency that renders it suitable to real-time monitoring of traffic and implement it on edge devices?

By addressing these questions, this study aims to contribute towards the development of intelligent transportation systems by offering an accurate, efficient, and scalable solution that is well-suited to the traffic conditions in Malaysia.

The rest of the paper is structured in the following way: Section II provides a review of the related work on the use of vehicle classification and counting approaches. Section III explains the methodology proposed that ought to be used such as data collection, preprocessing, and model design. In section IV, there is a discussion on and results of the experiment. Lastly, Section V closes the study and provides the possible directions of future study.

2. LITERATURE REVIEW

Several approaches have been explored for automated vehicle detection, counting, and classification. Early work often relied on infrastructure sensors or classical image processing. Mechanical methods redefine traffic monitoring using sensors embedded in road surfaces or roadside infrastructure. In these methods technologies like piezoelectric sensors, inductive loops, or pneumatic tubes detect features of moving vehicles, conducting vehicle counting and classification based on size and composition [5]. Accuracy, reliability, and efficiency improve when mechanical methods are used instead of conventional approaches. These frameworks minimize human intervention and there is real time data to make data driven decisions. Traffic monitoring is improved by Mechanical Techniques. The mechanical vehicle classification process is illustrated in Figure 4, where signals from piezoelectric, acoustic, and other sensors are used to make mechanical measurements, producing signal data that indicates mechanical vehicle types and counts.

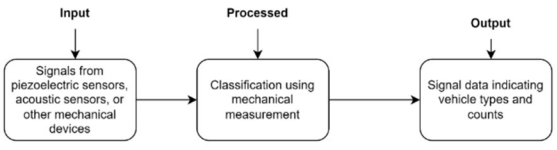

Figure 3: Mechanical Methods Process Flow Diagram

Piezoelectric sensors can convert mechanical stress to electrical signals. However, sensitivity to temperature and surface conditions of the vehicle make the system highly dependent on precise calibration, and they have shown high rates of accuracy in spot counting of vehicles [6]. Similarly, sound patterns coming out of the engine and the tire are used by the acoustic sensors to identify and classify vehicles. However, they are helpful, where visual detection is poor, like tunnels [7]. Together, these sensors constitute a nonintrusive complement to other methods of detection. While Mechanical methods provide accurate data about the detection and counting of vehicles they may not be able to classify the vehicles with fine stability [8]. Recently, much research interest has been shown in merging mechanical sensors with deep learning approaches to make the system much more accurate and efficient for vehicle classification. According to Neupane (2022) the development of hybrid systems that can provide insights into the dynamics of traffic and reduce the limitations of individual approaches by combining the strengths of both mechanical and deep learning methods [9].

Artificial Intelligence, from its subfield Machine Learning, has emerged as a powerful set of technologies in the transformation of a variety of sectors, including transportation management. This part outlines the applications of AI and ML techniques in traffic analysis, vehicle detection, and classification. Especially, it considers the role that deep learning techniques are now performing in changing the automatic systems for vehicle detection, classification, and counting. For trained traffic management decisions, AI processes data derived from cameras, sensors, GPS and mobile applications [10]. Traffic patterns are identified, and congestion can be predicted by AI methods. Then they can optimize traffic flow and improve the efficient use of resources. Similarly, highly precise vehicle detection and tracking is performed in real time over video feeds using e.g. CNNs, as developed by [11]. With these methods, we have reliable detection in challenging conditions. AI enables detection based on multi-sensor data inputs.

Based on their characteristics, most importantly size and shape, deep learning algorithms categorize vehicles. Classifying with accuracy optimizes traffic flow and safety, as well [12]. Traffic management strategies are based on

this information. In recent years, deep learning methods have empowered the field of vehicle recognition: the ability to detect and classify vehicles from still images and video streams using complex neural network architectures [13]. According to Jagannathan (2021) Deep Learning Techniques for Vehicle Detection and Classification are sensitive to advanced neural network architectures that automatically identify and classify vehicles in images and videos [14]. Figure 5 shows the digital, AI based vehicle classification workflow where vehicle types, counts, and potential behaviors are provided using AI algorithms such as CNNs and RNNs, or data from mechanical counting gates.

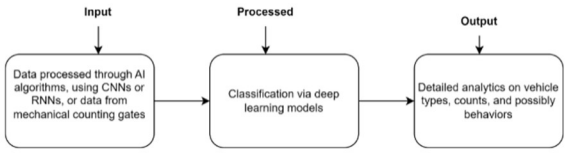

Figure 5: Artificial Intelligence/DL Methods Process Flow Diagram

The feature extraction and classification parts in CNNs are layers that are widely used in traffic monitoring and autonomous driving [15]. An important contribution is their ability to efficiently capture spatial hierarchies within visual data. Computationally efficient CNNs are applied to real time traffic monitoring systems and RNN is used to handle sequential data as it is used to track vehicles over time in video streams [16]. Their architectures for predicting vehicle positions are Long Short-Term Memory (LSTM) networks. In real time surveillance and self-driving vehicle navigation, applies RNN [17]. LSTM and RNNs function as the most used deep learning architectures for processing sequential data while boosting the analysis of time-series inputs [18]. Temporally dependent information identification capabilities make these models the choice for applications which require time-series forecasting as well as natural language processing and anomaly detection. DeepAR is an autoregressive recurrent network that utilizes LSTMs for probabilistic time-series forecasting, effectively capturing uncertainty [19]. Similarly, deep learning models for time-series forecasting remain relevant due to their continued use of LSTMs and RNNs despite new architecture advancements [20]. The time-series methods experience limitations because their high computational complexity combined with difficulties in processing extended series makes it necessary to seek different model solutions.

The extraction of spatial features with great efficiency makes Convolutional Neural Networks (CNNs) particularly 2DCNNs extremely effective for image recognition tasks. The application of 2DCNNs to numeric data from time-series or tabular datasets requires extensive adjustments before implementation. According to Wang (2016) one-dimensional data lacks optimal results when 2DCNNs apply directly to numeric time-series data. The need for efficient processing of sequential numeric information has generated 1DCNN architecture development for such specific tasks [21]. According to Ismail (2019) time-series classification reaches state-of-the-art performance levels when suitable CNN adaptations are applied [22].

Recently, 1DCNNs have gained widespread popularity as a robust alternative to traditional models such as LSTMs and RNNs on specific numeric datasets [23]. They offer good performance in extracting hierarchical patterns efficiently and computationally cheap when compared to other models, and they suit the tasks of time series classification, data analysis using sensors and other devices. Kiranyaz (2019) conducted a comprehensive survey of 1DCNNs, including their superiority on applications like ECG classification and industrial fault detection [24]. Similarly, Zhang (2020), proposes TapNet a multivariate time series classification model that achieves competitive performance under computational efficiency using 1DCNNs [25]. These studies furthermore highlight that 1DCNNs are emerging as a versatile and efficient numeric data analysis tool.

Table 1 provides an overview of vehicle classification tools categorized by method, including manual methods like tally count sheets and clickers, survey tools such as CCTV and ANPR, mechanical methods including inductive loop sensors and radar, and digital/IT tools like mobile apps with GPS data and traffic simulation software.

Table 1: Existing Tools for Vehicles Classification and Counting.

Category	Tool	Description
Manual Methods	Tally Count Sheets	Simple paper-based method where individuals manually count vehicles passing a specific point.
	Clickers	Handheld devices used by personnel to count vehicles by pressing a button with each vehicle passing.
Survey Methods	Closed-Circuit Television (CCTV)	Existing CCTV cameras can be used to analyze vehicle movements and counts at intersections or along roads.
	Automatic Number Plate Recognition (ANPR)	Systems that automatically capture and recognize vehicle number plates, often used for traffic monitoring.
Mechanical Methods	Inductive Loop Sensors	Embedded in the road surface, these detect vehicles passing over them and can provide data on traffic flow, speed, and vehicle counts.
	Radar	Measures vehicle speed and presence based on radio waves.
	LiDAR	Uses laser beams to detect vehicles and measure distance, often for more precise traffic monitoring.
(Digital/IT)	Mobile Apps and GPS Data	Apps installed on smartphones can collect GPS data anonymously to analyze traffic patterns and densities. GPS tracking of fleets or public transportation can also provide insights into traffic flow and vehicle movements.
	Traffic Simulation Software	Software programs like VISSIM, CORSIM, and Aimsun simulate traffic patterns based on inputs such as road networks, traffic lights, and vehicle types.

Although manual and mechanical algorithms offer simple vehicle counting and detection, they usually have low scalability, environmental sensitivity, and poor classification. AI-based methods, specifically the deep learning models, are more accurate and more automated, but may be computationally expensive and need large, well-labeled datasets. Even with these improvements, there is still a challenge related to processing time-series sensor data in an efficient manner, separating structurally similar vehicular types, and real-time performance. This paper will fill in these gaps by introducing a Deep learning model which can utilize hybrid datasets to enhance classification, counting accuracy, and computational efficiency.

3. PROPOSED METHODOLOGY

In addition to the exceptional performance in controlled conditions, the 1D-CNN suggested framework shows a high level of flexibility in cases when the model is tested on hybrid data reflecting a real-life situation. The model benefits from an efficient design that enables it to handle to address the problem of noisy, irregular, and incomplete data, which may commonly arise in real-life sensor deployment. The end to end design of the framework in terms of data collection and final evaluation indicates a holistic approach to the problems of the classical vehicle classification systems. The combination of low computational overhead, and fast inference makes this solution suitable to integrate into the intelligent transportation infrastructures and have a potential to be aimed at large-scale implementations in urban mobility systems planning and automated control systems of traffic.

3.1 Data Set and Preprocessing

The time sequence data was obtained from embedded piezoelectric sensors in road surface. The real time signals acquired from these sensors included voltage fluctuations (EMF) and force exerted by the passing vehicles on the vehicle structure. The recorded signals served as a primary data source for the separation of the separate vehicle types, which are then used as a criterion for selection of data for the model.

To make the model potentially more accurate, a thorough preprocessing strategy was used to ensure the data quality. It involved cleaning the data, removing inconsistencies and errors, normalizing the dataset to make sure range

of the feature were the same, and segmenting the signal into smaller meaningful sequences. The categorization of vehicle types was achieved by applying label encoding; therefore, the data was structured in a way that is good for model training and inference. The noise of data and the complexity of computation have been suppressed by these preprocessing techniques, and these are important preprocessing techniques which have prepared the data so that deep learning can perform analysis over that data.

This experimental scenario introduces a hybrid dataset that blends samples of real-world numeric data with synthetic data, maintaining the same range of 100,000 data points. The purpose of this dataset is to present a case of real-life problems in which data are not noise free, missing values are present and consistency is disturbed. The hybrid dataset tests the adaptability of models to real-world conditions, ensuring that their performance remains robust across diverse data distributions, it also helps in determining whether 1DCNN still has the edge in terms of efficiency in dealing with more complicated dataset.

This dataset employs all metadata fields from synthetic datasets and adds timestamp, vehicle category, axle count, weight, event type (approach or depart) and signal duration to its structure. Each timestamp and event type inclusion point provides better vehicle movement analysis to satisfy traffic monitoring requirements through enhanced dataset functionality. Real-world traffic system information made up the dataset's actual section whereas synthetic simulation outputs provided supplementary values to balance and complete missing real-data points. Hybrid dataset derives its format and structure from the published thesis and paper [26]. The dataset structure presents deep learning models with real-world challenges which enable them to become more resilient for actual deployment conditions. Table 2 shows below the format of the data used for the model.

Table 2: Format of Data

Field Name	Data Type	Description
Timestamp	Date Time	Time of vehicle detection event.
Vehicle Category	Categorical (Label Encoded)	Type of vehicle (e.g., car, truck, motorcycle).
Axle Count	Integer	Number of axles detected.
Weight	Numeric (Float)	Estimated/measured vehicle weight (e.g., in kg or tons).
Event Type	Categorical	"Approach" (a vehicle entering sensor range) or "Depart" (exiting).
Signal Duration	Numeric (Float)	Duration (in seconds) of the recorded sensor signal.
Voltage Fluctuations (EMF)	Numeric (Float)	Raw voltage signals from piezoelectric sensors.
Force Exerted	Numeric (Float)	Force (e.g., in Newtons) applied by vehicle on the road.

The hybrid data was proposed to eliminate the drawbacks of using just either purely synthetic or purely real-world sensor data. In real-world datasets, the variables and data are not always complete, noisy, and consistent depending on environmental and deployment conditions, and the synthetic datasets cannot be considered sufficient to reflect the diversity of the real traffic conditions. Integrating both sources means that the hybrid dataset represents a balanced view of a variety of operating conditions. The synthetic component guarantees abundant, controlled and highly distributed data to prevent data scarcity, and the real-world component provides natural variability, missing values, and noise that are characteristic of real-life deployment conditions. This unity makes the suggested model more robust and allows the trained 1D-CNN framework to generalize successfully and become more resilient to real-time traffic monitoring and large-scale integration into the intelligent transportation system.

3.1.1 Preprocessing data

The data are pre-processed with respect to several steps before the raw data are fed into the CNN model. This makes sure that the quality of the data fed is right and enhances the performance of the model. These steps become necessary for the preprocessing of data into a format from which it could learn optimally and make accurate predictions. The following sections detail each preprocessing step.

The first stage of the Preprocessing process includes Data cleaning where most of the noisy and irrelevance data in the dataset are either

eliminated or removed. Noise is that which is every other information that does not form the basis of training of a particular model, information which when incorporated into the training set can bias the model. Outliers which are not respecting the features of vehicle detection like sensor noises or any other disturbance not related to vehicle detection or objects of any kind, are omitted. This process will enable removal of undesired data from the dataset and rejection of incorrect data which is very essential in the training of a good model.

Normalization is the scaling of data into a common range and most often, the range that is used is 0 and 1 or -1 and 1 depending on the method used. This becomes necessary for consistency not only across type of vehicles but also when it comes to different levels or even pules of signals created by the sensors. Normalization decreases any possibility of model bias since through the normalization process, features used in the model are scaled in a consistent manner. This process helps to introduce more uniformity than the model now has the space to treat all the data input in a more consistent manner leading to more accurate and balanced performance estimation.

Segmentation separates the record that flows in parallel to the continuous time series and divides it into several parts which reflect several vehicle passages. The step of segmentation is vital for the purpose of separating unique events within the continuous stream of data and enables the model to study individual vehicle interactions as opposed to the entwined unified data stream. The splitting of data into smaller and more manageable parts can cause better generalization of the characteristics of each car passage with better performance of the proposed model at the tasks of classification and counting.

Label Encoding, in the ultimate step of preprocessing, the data transit from categorical types to numerical values is done by a process called label encoding. Since CNN models cannot identify raw categorical data, this encoding operation transforms categorical data into numerical form to be interpreted by the CNN model in this case, transforming the vehicle type label data that include car, truck and bus into a numerical form that is understandable by the model. But this is the only step of conversion for the convenience of the agreement between the dataset and the CNN model, to extract and to learn the categorical information captured.

3.2 1DCNN Model Architecture

The architectural framework of the method that is used to classify the vehicles and count them is 1D CNN model. As shown in Figure 6 it is a multi-layer architecture, and each layer has been designed to extract and subsequently process features from the time series data. Following is the layers' explanation.

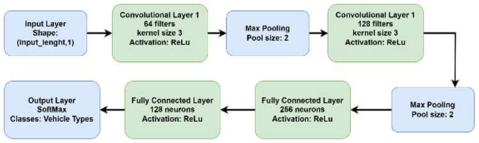

Figure 6: Architecture of 1D CNN Model

The input layer takes the time series preprocessed data. Every input sequence is the signal that the piezoelectric sensor gives out each time a vehicle traverses the sensing unit. The data input has the shape of (input_length,1), and 1 shows the dimension of the data where the input_length refers to the length of the time-series input. As such, this layer can be considered as the entrance of data flow for the model.

The Convolutional layers in the one-dimensional CNN model apply the convolution filters to extract all notable features from time-series input data. These prominent features will be discovered by several convolutional layers that allow the model to find a pattern leading to vehicle classification. The first convolutional layer employs filters which are small trainable weight matrices that move over the input sequence and 64 of them are used in this case. Each convolution is used to detect features in small portions of the time-series data. The output is 64 feature maps and each again represents unique features such as peak or movement profile which can have correlation with the movement of the vehicle above the sensor. The size of each kernel is kept constant to 3 steps to enable the model to identify the local features present in the input data. This small size of the kernel is helpful, because it allows focusing on the details which are refined – slight variations in the signal containing various kinds of vehicles.

Pooling layers prove very crucial in working of our proposed 1D CNN model by basically decreasing the dimensions of the feature maps that come out from the convolutional layer. This reduction is to make the model efficient and prevent overfitting which is a situation where a

model becomes complicated in a way that it tends to learn the training data set well but does an extremely poor job in doing so on new data sets. After every convolutional layer, a pooling size of 2 is used. This operation decreases the spatial dimensions of the feature maps by selecting from each pooling window the maximum value maintaining only the most notable features.

Fully connected (dense) layers represent a critical component of the neural network, aggregating the features learned by the convolutional layers to make final classification decisions. These layers are densely connected; thus, each neuron in that layer is connected to every neuron in the next or subsequent layer. This dense connectivity helps the model use the learned features in a proper way, and most of the features' combinations can be used in making the decisions. This layer comprises 256 neurons and employs the ReLU (Rectified Linear Unit) activation function to introduce nonlinearity in architecture. The reason for making it non-linear is to make the network capture as many relationships in the data as possible or put it in another way, it can capture complex patterns that linear models cannot capture. The second layer comprises 128 neurons with ReLU activation function in it. This layer produces a more abstract representation of the features extracted from data and, therefore, can be conceived of as a sort of 'preparation' for the last layer operating the actual classification.

The output layer is the last layer of neural network which essentially aims to classify the input data in one of several pre-specified categories of vehicles depending on whether they are cars, trucks or motorcycles among five classes. In this layer there are several neurons equal to the number of classes, where each neuron corresponds to a particular class. The output layer uses the SoftMax activation function that scales the output of the previous layer and maps them to probability values across the number of classes. This distribution enables the network to estimate the probability that the input of data belongs to the given category. The member of the input class that is most likely to belong to the input vehicle type is then determined to be the model's decision on the vehicle type. SoftMax function selects a network output that is distinguishable easily and ensures that the output is probabilistic since it recognizes the degree of certainty in the forecast made. This is particularly the case in multi-class classification, and that is why the output layer is of great significance under the probabilistic approach with a measure of accuracy and reliability of the model.

Table 3: Code for Layers of 1D-CNN

Layers	Code

Model Define	# Define the 1D CNN model
	model = Sequential ()

Input Layer	input_shape= (input_length, 1)))
Convolutional Layers	model.add (Conv1D (filters=64, kernel_size=3, activation='relu', # Second convolutional layer model.add (Conv1D (filters=128, kernel_size=3, activation='relu'))
Pooling Layers	# Max pooling model.add (MaxPooling1D(pool_size=2))
Fully Connected Layers	# Fully connected layers model.add (Dense (256, activation='relu')) model.add (Dense (128, activation='relu'))
Output Layer	# Output layer with softmax activation model.add (Dense (num_classes, activation='softmax'))

3.2.1 Rationale for Selecting 1D-CNN

The justification for selecting 1D-CNN over other potential deep learning architectures is presented below, highlighting its strengths in comparison with alternative models.

Table 4: Comparison of Deep Learning Models for Vehicle Classification

Model	Advantages	Limitations	Why 1D-CNN is Better	Reference
1D-CNN	Efficient for time-series; extracts hierarchical features; scalable; real-time capable	Limited to time-series, not image tasks	Optimized for sensor data, more efficient than others	(Shahid et al., 2022)
RNN	Captures temporal dependencies; good for sequential data	Vanishing gradient; high cost	1D-CNN avoids gradient issues, faster for sensor signals	(Hewamalage et al., 2021)
2D-CNN	Strong for images; learns spatial patterns	Heavy computation; poor for series	1D-CNN lighter, specialized for sequential data	(Chen & Ye, 2020)
LSTM	Learns long dependencies; avoids vanishing gradient	Slow, costly, risk of overfitting	1D-CNN faster and less prone to overfitting	(Kong et al., 2024)
Autoencoder	Learns features; good for dimensionality reduction	Needs fine-tuning; weaker in supervised tasks	1D-CNN better in supervised tasks, large datasets	(Yang et al., 2020)

3.3 Model Training

The training phase of a well-developed 1D CNN vehicle classifier was found to be a crucial step. During training phase, the preprocessed data goes for training the model is extracted and fed to the model to make its parameters, weights, and bias learn unique signature for every type of vehicle. This process is described in greater detail step by step below.

Since the model should generalize well on data it has never seen, the dataset will be split into three subsets, training, test and validation set. The training set, being the largest portion of the dataset, is used to teach the model to recognize vehicle patterns. It utilizes this data to modify its parameters within the model, for the purpose of identifying those patterns associated with various sorts of vehicle. Validation Set, after every iteration of training, the model is evaluated with the elements in the validation set. This gives information about the accuracy regarding the tweaking of certain hyper parameters such as the learning rate and the batch size. The early sign of overfitting can also be got using a validation set to make sure that performance is not too high on the training data only on the next data also. Test Set, this will be useful for the model's last assessment of where accuracy will be determined. The splitting methodology of the data is useful in ensuring that the model does not over fit during the evaluations and provides the model with an unbiased measure of its performance on unseen data.

As such the Categorical Cross-Entropy loss function calculates a measure of the difference between the probability distribution which has been predicted by the model and the actual one-hot encoded labels. One of the multi-class problems or vehicle type classification is particularly attracted towards the categorical cross-entropy. The formula for the loss is given as:

$$loss = -\sum_{i=1}^{N} \text{yilog(y^i)} \qquad (1)$$

Where:

y_i is the true label (1 for the correct vehicle type and 0 for others).

y^i is the predicted probability of the i^{th} class being correct.

That is a loss function which has high values If the predicted probabilities are far from the real labels. The model also tries to optimize the amount of loss, while training, with the aim of increasing the accuracy levels of its predictions.

Adam optimizer, also known as Adaptive Moment Estimation, is useful in updating the model weights and the bias through computation of gradients from back-propagation. Adam is used because it has an effective way of having the best of two popular optimization techniques combined. The one is Momentum which aids in increasing the rate of the downhill gradient descent and decrease oscillations. And the other is RMSprop which adjusts the learning rate with regards to the current deltas. This makes it possible for the optimizer to improve during training hence enhances its ability to perform better. Probably the most useful feature of Adam is the adaptive learning rate: the step size will be adjusted for each parameter individually; therefore, it is good for large datasets coupled with the complex models such as the 1D CNN. This in turn results in the above developments and moreover leads to faster convergence and better performance usually.

The model is then trained for 50 epochs where one epoch is the time it takes for the model to go through the entire training data set. In each epoch, the model adjusts internally for the parameters that fit the loss function; an early stopper is used to prevent the model from over learning the training data. If you remember, early stopping in its simplest form runs several checks on the model's performance as it is after each epoch's completion. To be precise, early stopping is conducted when the accuracy of the model on the validation set does not show further improvements in a specified number of cycles. This is what allows the model to generalize better, otherwise, all it will do is just memorize the training data.

The training data is split into batches, which, instead of using the entire dataset at one time, utilizes a batch size of 32 for the following reasons, Batch processing is beneficial in training as it helps in using very little memory as well as taking less time since the batches are smaller as compared to the full set. Small batch sizes also retain the update of gradients manageable, thus making the process of optimization smooth together with reducing the noise as compared to single sample update. This is achieved at a relatively smaller cost than it would have taken to undertake an overall update across the entire data set. The mini-batch approach stands somewhere in between from the point of view of computational procedures and fluctuations that occur during the learning process, which in any case accelerates convergence and provides higher accuracy for unforeseen data.

3.4 Evaluation Metrics

After the 1D CNN model has been trained, it is evaluated extensively for its performance in terms of correct classification of the vehicle type. This involves validating the model's performance on a test set that was not included in the training process. The evaluation shows exactly the accuracy in generalizing new data and that it is important to know how well the model performs in a real life situation. The parameters of evaluation are Accuracy, Precision, Recall and Confusion Matrix which provide an overall view of the efficiency and inefficiency of the model.

Accuracy shows the Total percentage, by which the accuracy of the model indicates, was the model able to correctly predict vehicle types of the test set. Statistically, it is arrived at by Simply dividing the number of accurate positive results by the coupled accurate negative results and this is then divided by the summation of all actual positives and negatives no matter which model was used. In other words, accuracy is how many of the test data set were appropriately classified into the test vehicles set correctly. Even though accuracy can sometimes be valuable, it does not show efficiency in cases when there is a significant distinction of classes, and several types of vehicles represent more of a dataset than others. For this reason, several other metrics are also used to include precision as well as recall.

$$Accuracy = \frac{Correct\ Predictions}{Total\ Predictions} \quad (2)$$

Precision, on the other hand, is a finer measure which describes the accuracy of the model when it predicts positive samples. In other words, it measures the proportion of the total number of positive forecasts that are 'true' and hence has met the conditions of a good forecast. Therefore, if the model has high precision in the context of classifying vehicle, this implies that if the model guesses that a certain type of vehicle is likely to be of a certain type, then such assumption

holds a lot of truth. This is especially so when this cost is going to be extremely high; it becomes important not to leave too many wrong classifications.

$$Precision = \frac{True\ Positives}{True\ Positives + Fal\quad Positives} \quad (3)$$

Recall or Sensitivity is the ability of the model to select true positive results among the real positives out there in the field. In vehicle classification, recall would reveal to us how accurately the model has classified a specific type of vehicle. A high recall implies that most of the vehicle of a certain class are extracted but at the same can contain many false positives. Here, recall is important most especially if the cost associated with a true positive for instance, the failure to classify a vehicle as one – is more compared to the cost of a false positive.

$$Precision = \frac{True\ Positives}{True\ Positives + F\quad Negatives} \quad (4)$$

The confusion matrix is another important feature that is widely used for the purposes of visualizing the model's performance for vehicles of several types. It is the table that shows the number of correct and incorrect what about each class of the model. The rows of the matrix contain the actual vehicle types while the columns contain the predicted vehicle types. The individual cells within the matrix represent the number of predictions that fall under a particular category; true positive, false positive, true negative, and false negative. This way of using visualization enables one to determine which vehicle types are being confused with others to pinpoint certain changes they need to make to the model.

For instance, while working with a confusion matrix, knowledge is gained on how the model frequently confuses a truck with a bus, a direct route towards further fine-tuning the model. It enables one to determine where in the model the loss is high or low, and therefore a better understanding of model performance compared to just the accuracy.

4. RESULTS

Diverse metrics for 1D Convolutional Neural Network (1DCNN)- based framework for vehicle classification and counting like classification accuracy, precision, recall, F1 score, Mean Average Precision (mAP), Confusion Matrix, counting accuracy, inference speed and computational efficiency are evaluated. Results indicate high accuracy and reliability of the framework in classifying and counting vehicles.

4.1 Framework Performance Overview

This proposed framework reaches an overall test accuracy of 0.99, which demonstrates its ability to correctly classify and count the vehicles within the various categories. The high accuracy level demonstrates that the model has learned the complex patterns needed to effectively differentiate between different vehicle types. In real world context the model generalizes well as it has a 0.99 unseen data accuracy and proves to be robust and adaptable to unseen data. With inference speed being 17.69 seconds, the model is quick enough to serve its purpose in real time intelligent transportation applications. Furthermore, the framework utilizes 84.4% of the memory amount, which is an ideal trade-off between computational resource usage and model performance.

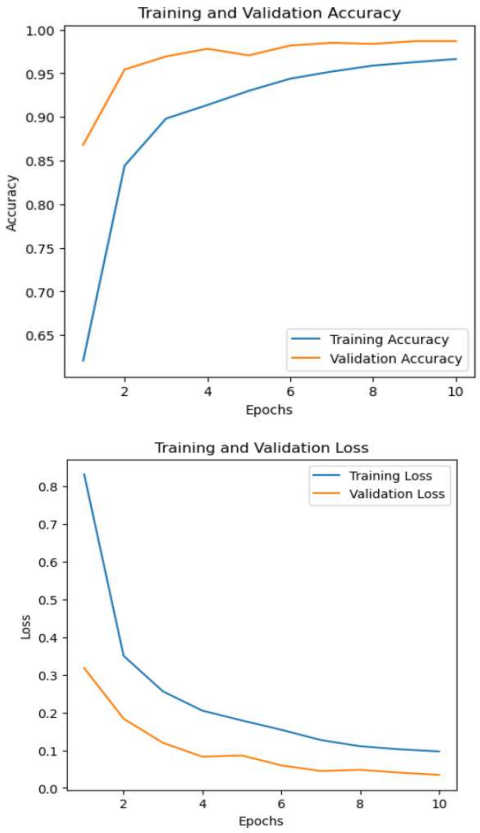

Figure 7: Framework's Model Accuracy and Loss

In Figure 7, the graphs presented show the training and validation performance of a machine learning model in the form of 10 epochs. These visualizations give indication of how well the model learns from the training data and what level of generalization is present for the test data that is not seen by the model during training. The left figure shows training and validation accuracy over time, the right figure training and validation loss.

The accuracy graph has its x-axis labelled as number of epochs and y-axis labelled with achieved accuracy by the model. The blue line, the training accuracy line starts low and slowly increases as the model learns the patterns in the dataset. After the final epoch, accuracy reaches above 95%, implying that the training data has been well fitted by the model. As shown in the orange line, the validation accuracy has a higher initial value and rises quickly up to about 99%. This indicates that the model has high generalization performance, since it can accurately classify on the validation set. However, the small difference between the accuracy of training and validation accuracy indicates a low amount of overfitting.

Additional insights regarding the model's learning 'feel' can be obtained through the right plot (training, validation loss). The number of epochs is plotted on the x-axis, while the corresponding loss values are shown on the y-axis, which indicates the measure of the prediction error. In the blue line, the training loss initial value is high but decreases steadily as the model adjusts its parameters. However, the fact that it is decreasing this steadily means that the model is learning well. In the same manner, the orange line plot shows that the validation loss decreases as well and is lower than the training loss consistently. Training and validation loss are in a small gap showing the model is well regularized and does not overfit. But validation loss slightly fluctuated in the late epoch. It means that the validation data is not perfectly fixed.

Overall, these graphs show that the model has learned the training data well while having very strong generalization. While there is a tiny signal of overfitting, it is not substantial because the validation accuracy stays high, and the validation loss doesn't rise. To further improve generalization, strategies can be used such as dropout, early stopping, data augmentation etc. Since the loss trends indicate that the model is still learning, we can continue training the model until diminishing returns on learning. The validation metrics are stable, which means the model is sufficiently trained and performs well on both the training and validation sets.

These research results are important for understanding the model's learning behavior and performance. The accuracy graph is strengthened by strong generalization capability whereas the loss graph shows learning with small overfitting. Through these trends, it can be said that the model is well trained and is applicable to practical use.

4.2 Classification Metrics Evaluation

Precision, recall and F1-score are used to evaluate the classification module of the framework for several vehicle categories. The developed model is perfect in terms of classification accuracy across all the classes with precision, recall and F 1 score of 0.99. This is due to the high classification accuracy achieved from 1DCNN extracting meaningful temporal and spatial features from input data and thus vehicles can be robustly identified.

The best classification performance of the vehicle categories is obtained by motorcycles, cars, and vans that achieve a perfect F1-score of 1.00. This indicates that the model achieved perfect classification for these vehicle types, with no false positives or false negatives. The confusion matrix shows that there is a small number of slight misclassifications in buses & truck categories, due to that 16 truck instances are wrongly classified as buses and 3 vans misclassified as buses. Visual and feature similarities of these categories may lead to this minor confusion on the synthetic datasets. Table 5 shows the values achieved while training the model.

Table 5: Classification of Vehicles Parameters

Vehicle Type	Precision	Recall	F1-Score	Support
Bus	0.96	0.99	0.98	400
Car	0.99	1.00	1.00	400
Motorcycle	1.00	1.00	1.00	400
Truck	0.99	0.96	0.98	400
Van	1.00	0.99	1.00	400
Overall	0.99	0.99	0.99	2000

Despite the model being extremely accurate in the majority of classes, a more detailed examination demonstrates that some of them, specifically buses and trucks, were more likely to be misclassified. This is due to similarities in structures and features of these two types of vehicles since both produce similar signal patterns as they drive over piezoelectric sensors. Moreover, these confusions can also be caused by overlapping signals, traffic density, minor differences in vehicle weight distribution, and some other factors. Although the net effect on the performance of classification is slight, the results from these studies demonstrate the need to make feature extraction more refined and consider more discriminative features in further research.

To also examine the strength of the framework, we also plotted Receiver Operating Characteristic (ROC) curves and respective Area Under the Curve (AUC) of each vehicle type. The ROC-AUC analysis gives a better understanding of trade-off between the true positive and false positive rates considering all thresholds. The five classes had an AUC value near to 1.0, which proves the high discriminative ability of the model proposed. The AUC values of trucks (0.97) and buses (0.96) were however somewhat less than that of cars, motorcycles, and vans (all between 0.99-1.00) which corresponds with the confusion between the two categories in the confusion matrix. The Figure 8 below shows the values.

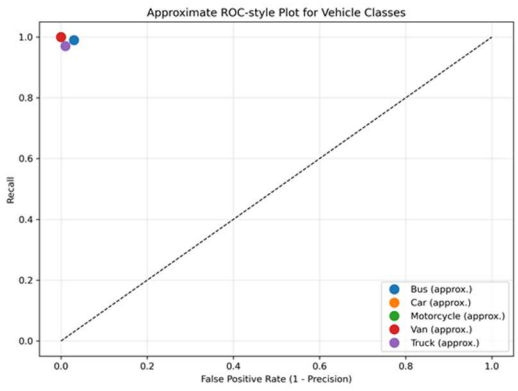

Figure 8: ROC Curves for Vehicle Classification

4.3 Detection and Counting Metrics Evaluation

Finally, the model is further analyzed on performance of the detection and counting with Mean Average Precision (mAP), confusion matrix and counting accuracy. We find that our model achieves a mAP of 0.98 with fine performance in detecting and differentiating vehicles under various traffic conditions. With a high mAP score, the model also positively localizes vehicles in complex environments.

A detailed confusion matrix was constructed for vehicle counting which gives insight into whether the framework has detected and counted vehicle sufficiently accurately in dynamic traffic conditions. This provides an estimate of the vehicles counted correctly and the miscount, due to occlusions, overlapping objects, etc. There were minor variations in buses and trucks, but most vehicle categories were counted with high precision. For instance, the actual number of buses was 400, which the model estimated to be 413, a slight overestimation. Similarly, trucks were also slightly under counted with 387 detected against the actual count of 400. This suggests that these variations might be because the model has trouble distinguishing between vehicles that are closely packed or overlapping, but this could be improved by refining the detection algorithm as shown in table 6 and 7 below.

Table 6: Actual Values Detection and Counting Metrics Performance

Metric	Value
Bus Count (Actual)	400
Truck Count (Actual)	400
Car Count (Actual)	400
Motorcycle Count (Actual)	400
Van Count (Actual)	400
Total Vehicles (Actual)	2000

Table 7: Predicted Values Detection and Counting Metrics Performance

Metric	Value
Bus Count (Predicted)	413
Truck Count (Predicted)	387
Car Count (Predicted)	403
Motorcycle Count (Predicted)	400
Van Count (Predicted)	397
Total Vehicles (Predicted)	2003
Mean Average Precision (mAP)	0.98
Counting Accuracy	0.99

Furthermore, another metric to measure the effectiveness of the framework for counting accuracy is to determine the number of vehicles being accurately counted. With a counting accuracy of 0.99, the model correctly counts the number of vehicles that are detected always. The high level of accuracy of counting confirms the reliability of the model as a vehicle analytics and real time traffic monitoring model and guarantees that those who wish to perform decision making based on the data. The results presented here indicate the good robustness of the 1DCNN based framework for vehicle detection and vehicle counting with very low error and allow it to be applied in real world intelligent transportation systems.

The results demonstrate that the proposed 1D-CNN achieved 0.99 classification accuracy across five vehicle classes, thereby addressing RQ1 by confirming the effectiveness of using piezoelectric signals for vehicle classification.

4.4 Performance Analysis on Unseen Data

This was used to further validate the robustness of the model by evaluating its classification and counting performance on unseen data of 10,000 vehicle instances. In the case of real-world conditions, the model has an overall accuracy of 0.99 for vehicle identification and counting. The classification report for unseen data for vehicle categories indicates precision, recall and F1-scores of near 1.00 which proves the accuracy of the model in detecting and classifying vehicles correctly. Table 8 shows the values of these metrics for classification performance below.

Table 8: Classification Performance on Unseen Data

Vehicle Category	Precision	Recall	F1-Score
Bus	0.97	0.99	~0.98
Car	1.00	1.00	1.00
Motorcycle	1.00	1.00	1.00
Van	1.00	1.00	1.00
Truck	0.99	0.97	~0.98
Overall Accuracy	0.99		

The model achieved 0.97 precision and 0.99 recall for each of the five vehicle categories with a slight tendency to sometimes misclassify a few bus instances. The model also accurately (with perfect precision and recall values of 1.00) classifies cars, motorcycles, and vans such that none were misclassified. For the truck category we had a precision of 0.99 and a recall of 0.97 which indicates that a few misclassifications would happen as a small number of trucks would be misclassified as buses as it happened for the training data as well.

Finally, the confusion matrix for unseen data details those misclassifications. We correctly classified 1,974 out of 2,000 bus instances and misclassified 26 trucks as buses. Among the truck category, 1,943 were correctly identified, and 57 were misclassified as buses, which shows the tendency of misclassification of these two related classes. All other categories (cars, motorcycles, vans, etc.) were perfectly classified, except for a few misclassified (8 vans that were misclassified as trucks). This indicates that the model performs extremely well, but refinement on feature tuning or data augmentation is possible to distinguish buses and trucks even better in our complex scenario as the values of Confusion matrix are shown below Table 9.

Table.9: Confusion Matrix for Unseen Data

Actual → Predicted	Bus	Car	Motorcycle	Van	Truck
Bus (2,000)	1,974	0	0	0	26
Car (2,000)	0	2,000	0	0	0
Motorcycle (2,000)	0	0	2,000	0	0
Van (2,000)	0	0	0	1,992	8
Truck (2,000)	57	0	0	0	1,943

In terms of vehicle counts, the model was extremely accurate to the degree that there were only extremely minor deviations to actual vehicle counts. The actual numbers of buses predicted were slightly higher at 2,013 instead of 2,000; and of trucks slightly lower at 1,943 as opposed to 2,000. The counting mechanism was effective as other vehicle categories such as cars, motorcycles, and vans had near exact predicted counts. The mean average precision (mAP) of 0.98 validated the strong detection capabilities of the framework as it was making sure that vehicles were correctly localized and classified.

The model trained on the hybrid dataset outperformed the real dataset, confirming that

combining real and synthetic data improves robustness under varying traffic conditions, thus answering RQ2.

4.5 Computational Efficiency And Real-Time Feasibility

However, in the real-world application of the proposed framework, computational efficiency is indispensable. A speed of inference of 17.69 seconds guarantees that the model will process traffic data near real-time and is feasible for integration into intelligent transportation systems. With 84.4% memory utilization, the model can utilize computational resources within reasonable overhead on available number of cores and thereby enables scalability across different hardware environments. It has been designed in such a way that it can be deployed on cloud based as well as edge computing platforms, which makes it even more suitable for use in the real time traffic monitoring and automated surveillance systems.

The proposed framework requires minimal computational resources and can operate in real-time on edge devices, thereby answering RQ3.

4.6 Limitations And Assumptions

Although the given 1D-CNN framework shows decent results in the vehicle classification and calculation tasks, a number of drawbacks should be admitted. To begin with, the accuracy of the model relies greatly on quality of data that is received by the piezoelectric sensors that are implanted in the road. Such sensors are effective yet are prone to the environmental variables including the temperature changes, degradation of the surfaces, alignment of the sensors, which can corrupt the signal with noise or inaccuracy. In addition, even though the performance assessment revealed excellent figures of excellence in the framework, there still exists a slight but significant vulnerability in its performance where there is misclassification between structural similarity between various vehicle classes, especially trucks and buses. This ambiguity can be caused because there is overlapping character of signals in the time-series data. The dataset that is trained also has some real world and synthetic data. Even though such hybrid method makes the model robust, the synthetic data does not represent the complexity of the real-life traffic and its unpredictability. Another assumption made by the study is of a rather steady traffic flow, with cars passing the sensor one after another and not in severe traffic congestion. The model used in a chaotic or congested traffic scenario will be difficult in real-life conditions. The model will not be able to find the difference and count vehicles correctly. Last but not least, the category of vehicles, trailers or construction equipment, are rare or not in the current dataset, which may narrow the potential applicability of the model to other contexts.

5. DISCUSSION

The 1D-CNN framework proposed attained high accuracy (0.99) in classification and counting of vehicles, showing that it is efficient with piezoelectric sensor signals. There were no major misclassifications, and minor misclassifications were between motorcycles and cars and vans because of the similarity in signals. The hybrid data enhanced robustness and synthesized synthetic data thus allowing the model to deal with real world variation.

The computer efficiency of the framework and its real time capability reliability makes it applicable in intelligent transportation applications. The weaknesses are reliance on sensor quality, low frequency of vehicle types, and efforts in extreme congestion. Future research may find more sensors, adaptive learning and more complex traffic situations to further increase accuracy and scalability.

6. CONCLUSION

The research makes a contribution to intelligent transportation systems by showing that 1D-CNN models can effectively classify and count vehicles based on piezoelectric sensor measurements, which is a scalable and computationally efficient solution to provide real-time traffic monitoring in cities.

The proposed framework was evaluated using a hybrid dataset, enabling controlled benchmarking and assessment of generalization capabilities. It is found that the model reaches a classification accuracy of 0.99 which proves its ability to distinguish different vehicle categories. For vehicle classification, the precision, recall, and F1 scores are always over 0.99, showing almost perfect performance.

In terms of presence or vehicle counting, mAP metric reached 0.98, which indicates the framework's ability to recognize and count

vehicles under distinct traffic circumstances. With deviations equally distributed above and below the expected values, the counting accuracy of 0.99 indicates minimal error between expected and delivered vehicle count. Because trucks and buses share similar structural features, a minor misclassification was observed in these vehicles, but the model performed extremely well on the rest of them.

The framework was found to be sufficiently computationally efficient as a 17.69 second inference time was observed. An optimal utilization of the resources of 84.4% provides guarantee to the memory utilization which facilitates the deployment of the model on cloud as well as edge computing platforms. Finally, the framework's ability to generalize the unseen data with 10,000 vehicle instances was confirmed through the performance on the data that remained unseen.

6.1 Future Research Directions

Going forward, a number of directions of future research can be pursued in order to increase the performance, extensibility, and flexibility of the proposed system. Another potential area is mixing multi-sensor fusion approaches, where data associated with a variety of sources can be used (LiDAR, radar, or acoustic sensors are all open here). This would be complementary information to reduce the fact that vehicles having similar profiles might be misclassified to increase robustness. Moreover, to allow application in real-life scenarios resource-continued locality such as edge devices at the roadside, there must be model optimization strategies to minimize computation overhead whilst maintaining accuracy e.g. pruning, quantization, or knowledge distillation. It is also necessary to deal with a constraint associated with traffic jams and car obstructions. The ability to easily determine and number the vehicles during a heavy and congested traffic situation using the algorithms would greatly increase the usefulness of the system. Furthermore, enforcing the adaptive or ongoing learning rules may enable the model to adapt with time and experience the new traffic patterns and new types of cars without the need to re-train the model completely. Another important subsequent step would be an expansion of the experimental validation to the bigger-scale real deployments of the model into different environmental factors, geographical, and infrastructural settings to examine the generalizability and robustness of model. And, in the last, it would be possible to incorporate the system into larger smart city systems to provide dynamic traffic signal control, congestion forecasting, and real-time data-driven infrastructure planning, so it would be an asset to contemporary intelligent transportation system.

REFERENCES

[1] H. Daim, "Vehicle Registration Statistics in Malaysia," Malaysia Automotive Association, Malaysia, Feb. 2023.

[2] K. Gkiotsalitis and O. Cats, "Public transport planning adaption under the COVID-19 pandemic crisis: literature review of research needs and directions," Transport Reviews, vol. 41, no. 3, pp. 1–19, Dec. 2020.

[3] Y. Chen and W. Hu, "Robust Vehicle Detection and Counting Algorithm Adapted to Complex Traffic Environments with Sudden Illumination Changes and Shadows," Sensors, vol. 20, no. 9, p. 2686, May 2020.

[4] C. Y. Chen, J. H. Lee, and M. Wong, "Vehicle Classification Using Image Processing," Transportation Research Part C: Emerging Technologies, vol. 48, pp. 361–373, 2014.

[5] P. Burnos, J. Gajda, P. Piwowar, R. Sroka, and T. Żegleń, "Measurements of Road Traffic Parameters Using Inductive Loops and Piezoelectric Sensors," Metrology and Measurement Systems, vol. 14, no. 2, pp. 187–203, 2007.

[6] H. Shokravi, F. Bakar, and A. Manaf, "A Review on Piezoelectric Sensors in Traffic Monitoring," Sensors and Actuators A: Physical, vol. 315, Article ID 112211, 2020.

[7] R. Rathod and A. Sharma, "Vehicle Classification Using Acoustic Sensors," Procedia Computer Science, vol. 170, pp. 880–887, 2020.

[8] C. Xu, Y. Wang, X. Bao, and F. Li, "Vehicle Classification Using an Imbalanced Dataset Based on a Single Magnetic Sensor," Sensors, vol. 18, no. 6, p. 1690, May 2018.

[9] B. Neupane, S. Basnet, and K. Thapa, "Hybrid Systems for Traffic Management Using Mechanical Sensors and AI," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 4, pp. 3678–3689, 2022.

[10] M. S. Akhtar and S. Moridpour, "Traffic Flow Prediction and Management Using AI: A Review," Journal of Transportation

Technologies, vol. 11, no. 2, pp. 114–128, 2021.

[11] S. Tak, H. Kim, and J. Lee, "Real-Time Vehicle Detection and Tracking Using Convolutional Neural Networks," Journal of Real-Time Image Processing, vol. 18, no. 2, pp. 245–260, 2021.

[12] H. Gholamhosseinian and C. Seitz, "Deep Learning for Real-Time Vehicle Classification," Sensors, vol. 21, no. 14, p. 4682, 2021.

[13] M. N. Kadhim, A. H. Mutlag, and D. A. Hammood, "Vehicle detection and classification from images/videos using deep learning architectures: A survey," AIP Conference Proceedings, vol. 3232, no. 1, p. 020034, 2024.

[14] A. Jagannathan, P. Kumar, and R. Mehta, "Deep Learning Techniques for Vehicle Detection and Classification: A Survey," IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 7, pp. 3945–3964, 2021.

[15] S. Dabiri, A. Heidari, and L. Chen, "Convolutional Neural Networks for Traffic Monitoring," Sensors, vol. 20, no. 10, p. 2821, 2020.

[16] I. Anjum and Q. Shahab, "Vehicle Tracking Using LSTM and RNN in Real-Time Video Streams," IEEE Access, vol. 11, pp. 2231–2244, 2023.

[17] H. Li, M. Zhang, and L. Zhao, "A Deep Learning Model Using LSTM for Real-Time Vehicle Surveillance," Pattern Recognition Letters, vol. 152, pp. 27–34, 2022.

[18] Y. Kong, "Unlocking the Power of LSTM for Long Term Time Series Forecasting," arXiv preprint arXiv:2408.10006v1, 2022.

[19] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks," International Journal of Forecasting, vol. 36, no. 3, pp. 1181–1191, 2020.

[20] B. Lim and S. Zohren, "Time-Series Forecasting with Deep Learning: A Survey," Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 379, no. 2194, Article ID 20200209, 2021.

[21] Z. Wang, W. Yan, and T. Oates, "Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline," in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Vancouver, Canada, pp. 1578–1585, Jul. 2016.

[22] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. A. Muller, "Deep Learning for Time Series Classification: A Review," Data Mining and Knowledge Discovery, vol. 33, no. 4, pp. 917–963, 2019.

[23] A. P. Wibawa, A. B. P. Utama, H. Elmunsyah, U. Pujianto, F. A. Dwiyanto, and L. Hernandez, "Time-series analysis with smoothed Convolutional Neural Network," Journal of Big Data, vol. 9, no. 1, Apr. 2022.

[24] S. Kiranyaz, T. Ince, O. Abdeljaber, and M. Gabbouj, "1D Convolutional Neural Networks and Applications: A Survey," Mechanical Systems and Signal Processing, vol. 151, Article ID 107398, 2021.

[25] X. Zhang, Y. Zheng, and Q. Sun, "TapNet: Multivariate Time Series Classification with Attentional Prototypical Networks," Proc. AAAI Conf. Artif. Intell., vol. 34, no. 4, pp. 6845–6852, 2020.

[26] R. Tariq, "Precision in Traffic Monitoring: Harnessing Piezoelectric Sensors for Tire Analysis and Vehicle Classification," [Online]. Available: https://d.lib.msu.edu/etd/51532. Accessed: Jul. 8, 2025.

[27] K. Bandara, C. Bergmeir, and H. Hewamalage, "LSTM-MSNet: Leveraging forecasts on sets of related time series with multiple seasonal patterns," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 4, pp. 1586–1599, Apr. 2021.

[28] Y. Chen, "Robust vehicle detection and counting algorithm adapted to complex urban traffic conditions," Sensors, vol. 20, no. 9, p. 2686, May 2020.

[29] H. Yang, Y. Zhang, Y. Zhang, H. Meng, S. Li, and X. Dai, "A fast vehicle counting and traffic volume estimation method based on convolutional neural network," IEEE Access, vol. 9, pp. 150522–150531, 2021.

[30] A. Kong, "A comprehensive high-level automated driving assistance system with integrated multi-functionality," IET Smart Cities, 2024. [Online]. Available: https://doi.org/10.1049/smc2.12076