Edge Computing for Effective and Efficient Traffic Characterization

by

Asif Khan
B.Sc., University of Engineering and Technology, Peshawar, Pakistan, 2020

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

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

University of Victoria

Edge Computing for Effective and Efficient Traffic Characterization

by

Asif Khan
B.Sc., University of Engineering and Technology Peshawar, Pakistan, 2020

Supervisory Committee

Dr. T. Aaron Gulliver, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Zawar Hussain Khan, Co-Supervisor
(Department of Electrical and Computer Engineering)

ABSTRACT

Traffic flow analysis is essential to develop smart urban mobility solutions. Many advanced traffic flow monitoring solutions have been proposed but they employ only a small number of parameters. To overcome this limitation, an edge computing solution is proposed based on nine traffic parameters, namely vehicle count, direction, speed, and type, flow, peak hour factor, density, time headway, and distance headway. This solution is low cost, low power, low data bandwidth, and easy to install, deploy, and maintain. It is a sensor node comprised of an RPi 4, Pi Camera, Intel Movidius NCS2, Xiaomi MI Power Bank, and Zong 4G Bolt+. Pre-trained models from the OpenVINO Toolkit are employed for vehicle detection and classification, and a Centroid Tracking Algorithm (CRA) is used to estimate vehicle speed. The measured traffic parameters are transmitted to the ThingSpeak cloud platform via 4G. The proposed solution was field-tested for one week (7 h/day), with approximately 10,000 vehicles per day. The count, classification, and speed accuracies obtained were 79.8%, 93.2%, and 82.9%, respectively. The sensor node can operate for approximately 8 h with a 10,000 mAh power bank and the required data bandwidth is 1.5 MB/h. The proposed edge computing solution overcomes the limitations of existing traffic monitoring systems and can work in complex and heterogeneous environments.

Table of Contents

List of Tables

List of Figures

List of Acronyms

4G	4th Generation
AI	Artificial Intelligence
API	Application Programming Interface
BRT	Bus Rapid Transport
CPU	Central Processing Unit
CRA	Centroid Tracking Algorithm
CSI	Camera Serial Interface
CV	Computer Vision
CV2	Computer Vision 2
fps	frames per second
GHG	Greenhouse Gas
GPS	Global Positioning System
ID	Identification
IoVT	Internet-of-Video-Things
ITSs	Intelligent Transportation Systems
mAh	Milliampere Hour
mAP	mean Average Precision
MB	Megabyte
ML	Machine Learning
MySQL	My Structured Query Language
NCS2	Neural Compute Stick 2
OpenCV	Open Computer Vision
OpenVINO	Open Visual Inference & Neural Network Optimization
R-CNN	Region-Based Convolutional Neural Network
RPi	Raspberry Pi
SSDs	Single Shot Detectors
USA	United States of America
USB	Universal Serial Bus

USD	United States Dollar
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
VPU	Visual Processing Unit
WHO	World Health Organization
Wi-Fi	Wireless Fidelity
YOLO	You Only Look Once

Acknowledgement

I would like to express my deep gratitude to my Supervisor Dr. Thomas Aaron Gulliver and my Co-Supervisor Dr. Zawar Hussain Khan for their invaluable guidance, support, and expertise throughout this research. I am also very thankful to Dr Khurram Shehzad Khattak for his tremendous contributions, encouragement, and motivation during this research. Moreover, my sincere appreciation goes to my family for their unwavering encouragement and understanding. Lastly, I extend my thanks to all those who have directly or indirectly contributed to this work. Your support has been instrumental in the completion of this thesis.

Dedication

I dedicate this thesis to all those who have supported and inspired me throughout this academic pursuit. Your encouragement and belief in my abilities have been invaluable. This work reflects the collective support and guidance I have received, and I am grateful for the impact you have had on this journey.

Chapter 1

Introduction

Urbanization is expected to drive global economic growth in the coming decades by increasing productivity and reducing poverty. However, this growth is threatened by challenges to urban mobility including Greenhouse Gas (GHG) emissions and lost productivity due to road accidents and congestion. The road transportation sector accounts for 25% of worldwide fuel consumption and 29% of GHG emissions [1]. Traffic congestion also results in a significant reduction in productivity, with an average driver in the United States of America (USA) losing 36 h and United States Dollar (USD) 564 in 2021 [2]. Moreover, according to the World Health Organization (WHO), road accidents cause 1.3 million deaths and 50 million non-fatal injuries every year [3]. Therefore, it is imperative to develop innovative and effective solutions such as Intelligent Transportation Systems (ITSs) to mitigate these challenges and ensure efficient urban mobility.

1.1. Intelligent Transportation Systems

An ITS integrates technology into transportation infrastructure and vehicles to manage and improve traffic flow, enhance safety, and reduce congestion. Edge computing plays a crucial role in ITSs by processing data closer to the source and enabling real-time analysis of traffic conditions, vehicle movements, and infrastructure status. This allows for faster decision-making, reduced latency, and improved response to traffic problems, leading to more efficient and effective traffic management.

Edge computing enables the processing and analysis of data at or near the source of data generation, such as traffic cameras, sensors, and connected vehicles. This decentralized approach to data processing reduces the need to send all data to a centralized location for analysis, which can result in latency and delays in decision-making. By using edge computing, ITSs can provide real-time insights into traffic conditions, identify congestion or accidents, optimize traffic signal timing, and even support autonomous vehicles with immediate local data processing. This allows for faster and more precise responses to traffic issues, leading to improved traffic flow, reduced

travel times, and enhanced overall transportation efficiency. Edge computing in ITSs can also support Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) communication, enabling vehicles to exchange information with each other and with the surrounding infrastructure, further enhancing safety and traffic management. Overall, the integration of edge computing in ITSs plays a pivotal role in creating a more responsive and adaptive transportation system, ultimately contributing to solving traffic problems and improving the overall user experience.

1.2. Role of Computer Vision in Intelligent Transportation Systems

ITSs leverage Computer Vision (CV) to improve the safety and effectiveness of transportation systems. CV enables machines to understand and interpret visual data from the surroundings. Using CV, an ITS can identify and track vehicles, pedestrians, and other roadside elements, and use this information to enhance safety and efficiency. CV is instrumental in monitoring and regulating traffic flow and identifying traffic congestion, accidents, and other incidents in real time. It provides data to traffic management systems to improve traffic flow, reduce congestion, and enhance safety.

CV can detect pedestrians, alerting drivers to their presence, which improves both pedestrian and driver safety. In the realm of autonomous vehicles, CV enables vehicles to perceive and process information about their surroundings and take appropriate actions. Moreover, CV has the ability to recognize license plates for applications such as toll collection, parking management, and law enforcement. It can be employed to monitor road conditions to identify issues like potholes and cracks and provide valuable insights to prioritize road maintenance and repairs.

1.3. Research Context

ITS-based solutions are a promising means of improving road network efficiency. Detailed traffic data including vehicle count, speed and classification, flow, spatial/temporal densities, vertical/horizontal headways, road capacity, heatmaps, and trajectories are essential for traffic engineers to improve transport network management. Furthermore, these parameters can be employed in traffic simulation software [4,5] to aid urban planners in designing effective road networks.

Both intrusive and non-intrusive traffic monitoring systems have been developed. However, these solutions have limitations, including only measuring traffic count and speed, installation and

maintenance difficulties, and high costs [6]. With advancements in image processing techniques, roadside video can now be employed. While Internet-of-Video-Things (IoVT) solutions can be effective, the high bandwidth requirements for roadside video transmission to servers are a major limitation [6,7]. Image processing edge computing solutions have been proposed to overcome this problem. However, the computational power of devices such as Raspberry Pi (RPi) limits the ability to provide detailed traffic information. Existing edge computing solutions provide either count [1,8,9,10], count and speed [11,12,13,14], or count and classification [11,15–18].

1.4. Scope and Objectives

In this work, an edge computing solution is proposed to overcome the limitations of existing traffic monitoring systems. The objective is to accurately obtain vehicle count, speed, type, and direction, flow, peak hour factor, density, time headway, and distance headway. This is achieved using a sensor node composed of an RPi 4, Pi Camera V2, Intel Movidius Neural Compute Stick 2 (NCS2), Xiaomi MI 10,000 mAh power bank, and Zong 4G Bolt+. The pre-trained MobileNet-SSD model from the Intel Open Visual Inference & Neural Network Optimization (OpenVINO) Toolkit is employed for vehicle count and classification [19]. Vehicle speed is estimated using the CRA running on the RPi 4. The measured traffic parameters are transmitted to the ThingSpeak cloud platform using 4G. This data can be used for traffic flow analysis to aid urban planners in transportation network planning and management.

The remainder of this thesis is organized as follows. Chapter 2 provides an overview of related work in the area. Chapter 3 presents the architecture of the proposed system including the hardware components and software algorithms used. Chapter 4 gives some experimental results including the accuracy and reliability of the proposed system. Finally, Chapter 5 provides some concluding remarks and suggestions for future research.

1.5. Conclusion

This chapter presented the challenges to urban mobility such as GHG emissions, road accidents, and congestion, emphasizing the need for innovative solutions for efficient urban mobility. The role of ITSs was discussed, focusing on the integration of edge computing and its benefits in processing real-time traffic data, improving traffic flow, and supporting vehicle communication. The role of CV in improving safety, regulating traffic flow, and enabling autonomous vehicles was

discussed. The use of CV in monitoring road conditions and traffic incidents was also outlined. The research context was presented, highlighting the limitations of existing traffic monitoring systems and proposing an edge computing solution to overcome these limitations. The scope and objectives of the proposed edge computing solution were outlined, detailing the hardware components, software algorithms, and the use of traffic parameters to aid in traffic flow analysis for personnel such as urban planners.

Chapter 2

Related Work

The development of intelligent mobility solutions requires accurate real-world traffic data. CV-based approaches have been shown to be better than intrusive and non-intrusive sensor-based solutions [6]. However, their capabilities are limited, primarily due to a lack of computational resources. Existing edge computing solutions provide either vehicle count, count and classification, or count and speed as discussed below.

2.1 Vehicle Count

An edge computing solution based on an RPi and a web camera was presented in [9] which achieves a vehicle count accuracy of 83%. Vehicle count, road density, time headway, and vehicle emissions were obtained with the system in [1]. This solution uses an RPi 4 and Pi Camera V2 with four sensors to measure carbon monoxide, carbon dioxide, and particulate matter. A vehicle count accuracy of 86% was reported and the measured parameters were transmitted to the ThingSpeak cloud platform using the RPi Wireless Fidelity (Wi-Fi) module.

2.2. Vehicle Count and Classification

In [15], a system to count and classify vehicles at a highway toll booth was developed using an RPi B and Pi Camera [8]. In [10], an edge computing solution to count and classify vehicles was presented which employs an RPi 2, Pi Camera, and My Structured Query Language (MySQL) web server database [17]. In [16], an RPi B and Samsung smart security camera were used to transmit parameters to a remote web server for display. A vehicle count accuracy of 83% was obtained. In [18], an edge computing solution was developed using an RPi 2 and Pi Camera to count vehicles and classify them as small or large. The data is stored locally on the RPi 2 for archiving purposes. In [20], a real-time stereo vision system was presented to count vehicles and classify them as cars or small or big trucks. It employs an RPi 3B and Universal Serial Bus (USB) webcam and transmits the parameters to a local web server for display.

2.3. Vehicle Count and Speed

A solution using an RPi 2B+ and Pi Camera to count vehicles and estimate their speed was given in [12]. The Flask web framework was used to archive the parameters on an edge cloud server. In [21], an edge computing solution to count vehicles and estimate their speed was presented which uses an RPi 2 and Pi Camera. The effect of different frame sizes on the Central Processing Unit (CPU) and memory was examined. It was found that the CPU performance was not significantly affected by the frame size, but higher resolution frames required more memory. In [22], a system to count vehicles and estimate their speed was developed which used an RPi 3B and Pi Camera. The parameters were stored locally on the RPi, and a count accuracy of 100% and speed accuracy of 90% were reported. Another solution was proposed in [13] which used an RPi 3B and Pi Camera. All these edge computing solutions are limited by the RPi computing resources. Thus, an edge computing solution is proposed here to overcome this constraint. The advantages of the proposed solution are as follows.

- Existing edge computing solutions only measure two traffic parameters, either vehicle count and classification [15–18,20,23] or vehicle count and speed [5,12,13,21,22]. Conversely, the proposed solution can measure nine traffic parameters, namely vehicle count, speed, direction, and type, flow, peak hour factor, density, time headway, and distance headway.
- The proposed solution can classify six different types of vehicles, namely cars, buses, motorbikes, bicycles, and animal drawn carts (horse and cow). This is greater than the number of vehicle classes provided by existing solutions [15–18,20].
- The proposed solution can count and classify vehicles with an accuracy of 93%, which is better than the accuracy reported in previous studies [1,4,9,14,16,22].
- The proposed solution can count and estimate the speed of a wide variety of vehicles and pedestrians. This includes trams (trains), airplanes, and boats. This is because the detection model was trained on over 70 different objects, including these vehicles. Vehicle direction is also obtained. This makes the proposed system ideal for characterizing heterogeneous traffic behavior. Note that no other system provides the direction of vehicles.
- The proposed solution was designed considering cost, reliability, and scalability. The sensor node costs less than USD 300 and has a low power consumption of 1.2 A per hour.

Unlike previous systems, the proposed solution transmits the measured parameters to a cloud platform using 4G with a data bandwidth requirement of approximately 1.5 MB per hour.

2.4. Conclusion

This chapter presented the existing edge computing solutions for intelligent mobility, focusing on the limitations and capabilities of CV-based approaches. These include edge computing solutions that measure vehicle count, count and classification, and count and speed. Furthermore, the proposed solution to overcome the limitations of existing systems was introduced. This solution can measure nine traffic parameters, classify six types of vehicles, achieve a vehicle count and classification accuracy of 93%, estimate the speed of various vehicles and pedestrians, and provide the direction of vehicles. It is cost-effective, reliable, and scalable, which sets it apart from previous systems.

Chapter 3

System Architecture

The proposed edge computing solution for real-time traffic characterization is shown in Figure 1. It measures nine traffic parameters and transmits them to a cloud platform using the Zong 4G Bolt+. The system comprises three main components: (1) sensor node, (2) CV module, and (3) cloud platform.

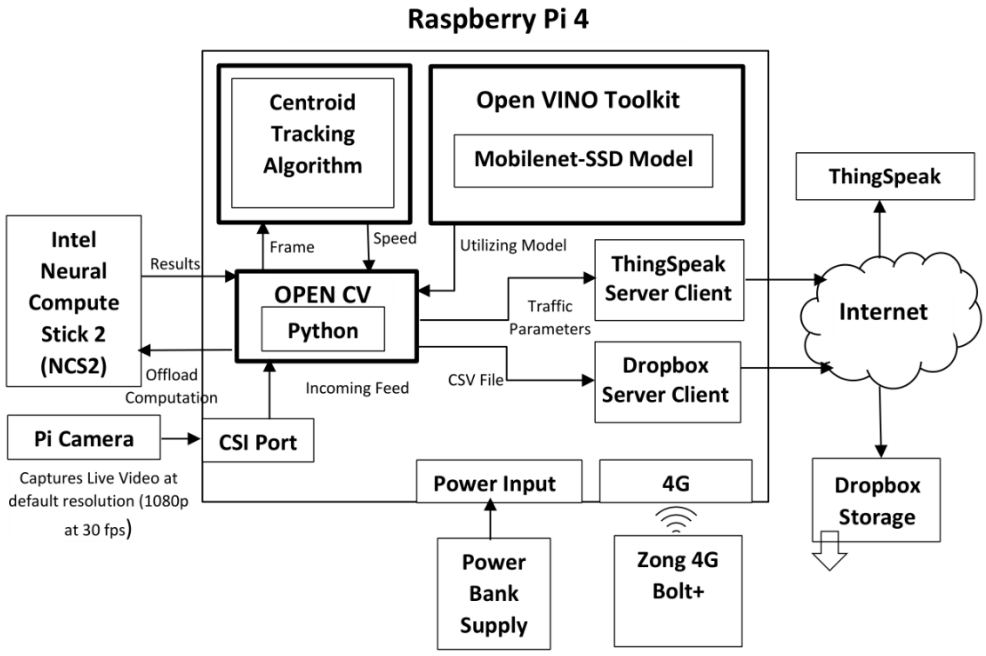


Figure 1: The architecture of the proposed system.

3.1. Sensor Node

The sensor node was fabricated using cost-effective but powerful hardware components. This included an RPi 4 (a low-cost Linux-based single-board computer), and a Pi Camera v2 connected via the Camera Serial Interface (CSI) port. It can capture roadside video at 20 frames per second (fps) with 1080p resolution. A 10,000 mAh Xiaomi Mi Power Bank 2 was used to provide

sufficient battery life for extended use. The sensor node was designed for edge deployment and only consumes 1.2 A per hour, measured using a Keweisi USB power tester. The measured traffic parameters are transmitted to the cloud platform using a Zong 4G Bolt+. This provided reliable and efficient communications with low complexity.

To overcome the computing limitations of a single board computer such as RPi [24], the proposed solution employs an Intel Movidius NCS2. This was designed to provide computation power to edge devices. It has a Myriad X Visual Processing Unit (VPU) and a dedicated hardware accelerator for Artificial Intelligence (AI) and CV applications [25]. This enables the offloading of complex computations from the CPU and consumes much less power.

3.1.1. Hardware Components

3.1.1.1. Pi Camera V2

The system uses an RPi Camera v2 which has high resolution and can capture video at 8 MP. The Pi Camera is attached to the RPi 4 through a CSI port. The system is configured through Open Computer Vision (OpenCV) to capture 1080p video at 30 fps. A script file in Python was written to capture real-time traffic video. OpenCV is used to perform image processing to retrieve traffic parameters and classify vehicles.

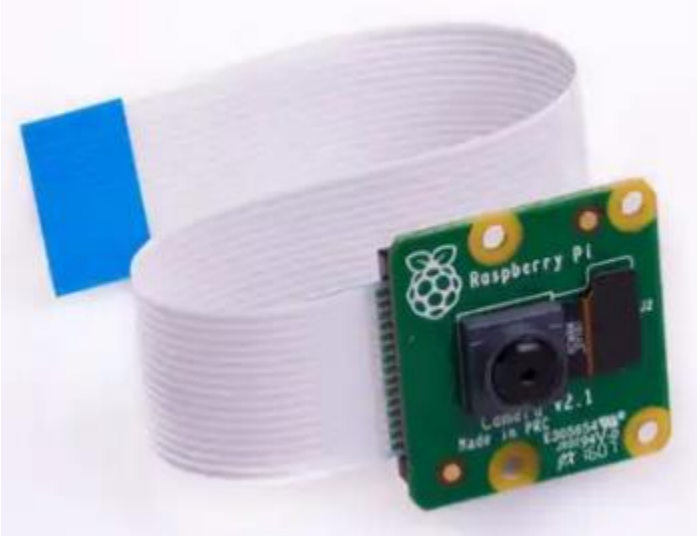


Figure 2: The Pi Camera V2 for video capture [35].

3.1.1.2. RPi 4

The RPi 4 shown in Figure 3 is a powerful Linux-based processing board. An RPi 4 is used in the proposed system to run OpenCV along with a neural stick to detect vehicles, estimate speed, and classify vehicles. The RPi 4 is powered by a power bank and connected to the internet via a 4G dongle device for remote use on the roadside. The choice of hardware was the latest RPi 4 model for efficient edge computation.

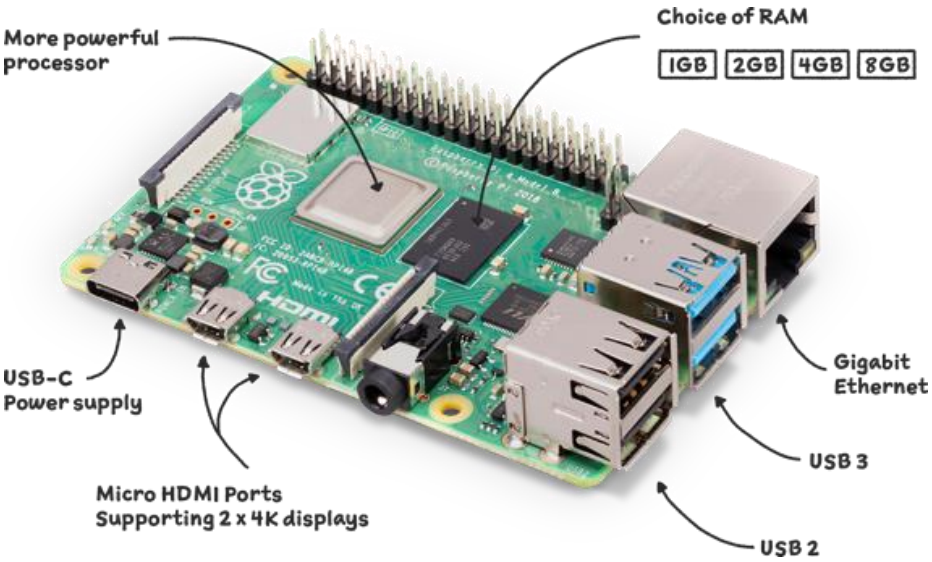


Figure 3: The RPi 4 processing board [36].

3.1.1.3. Intel Movidius Compute Stick 2

The Intel Movidius NCS2 was designed to provide computation power to IoT and edge devices for applications such as AI and CV. It uses the Intel Myriad X VPU and a dedicated hardware accelerator for neural networks. It can run neural network models with low power consumption. The Intel Movidius NCS2 shown in Figure 4 is used for computations in the sensor node.

Figure 4: The Intel Movidius NCS2 used for computation [37].

3.1.1.4. Xiaomi Mi Power Bank 2 and Zong 4G Bolt+

The Xiaomi Mi Power Bank 2 10,000 mAh power bank is used to power the sensor node and a portable Zong 4G Bolt+ is used to provide an uninterrupted remote connection. These devices are shown in Figure 5.

Figure 5: The Xiaomi Mi Power Bank 2 (left) and Zong 4G Bolt+ (right).

3.1.2. Software Components

3.1.2.1. Python and OpenCV

There are many Python libraries for image processing and CV. The proposed system uses OpenCV to capture video from the road. This video is obtained with the Pi Camera using the CV2 module in OpenCV.

3.1.2.2. MobileNet-SSD Model

Single Shot Detector (SSD) is a deep learning model to detect objects from an image or video. It comes with components known as backbone model and SSD head. The backbone model is used to classify objects using a trained image classification network. The SSD head is a convolution layer embedded with a backbone to create bounding boxes around the detected objects [14].

To provide a high frame rate, the SSD is used for object detection instead of You Only Look Once (YOLO) or Faster R-CNN [15]. Furthermore, due to limited resources on the edge such as power, computational capabilities, and cost, MobileNet, an efficient deep neural network, was selected. The MobileNet-SSD algorithm uses SSD and MobileNet to detect vehicles using a pre-trained Caffe model. This model was trained on over 70 objects but here it is used for cars, buses, motorbikes, bicycles, and animal drawn carts.

The MobileNet-SSD model is well suited for resource constrained devices such as RPi while providing a high fps and accurate real-time object detection [15]. This model is available in the OpenVINO Toolkit which contains numerous pre-trained models for the Neural Stick devices developed by Intel.

3.1.2.3. Centroid-Tracking Algorithm

The CRA in OpenCV is employed to track detected objects in video and images. This algorithm calculates the Euclidean distance between objects already detected and the centroid of objects detected in future frames [18]. It can be used with many object detection models.

The distance traveled by a vehicle is estimated using the distance between the centroids of the bounding boxes of the vehicle in two different frames. After estimating the distance traveled the speed is estimated as

$$v = \frac{s}{t} \quad (1)$$

where s is the distance and t is the time.

Figure 6 shows the CRA steps [38]. Step 1 demonstrates accepting a set of bounding box coordinates and computing the centroids. The centroids are the (x, y)-coordinates of the centers of the bounding boxes of the detected objects. As these are the initial bounding boxes, unique IDs are allocated to them. Step 1 is used for every subsequent frame to compute object centroids. However, instead of assigning a new ID to each detected object an association is determined between the new object centroids (yellow) and old object centroids (purple). This is accomplished in step 2 where the Euclidean distance (green arrows) between each pair of existing object centroids and new object centroids is calculated. Step 2 shows three objects detected in the image. The two closest pairs are the existing objects. The Euclidean distance between each pair of original centroids (yellow) and new centroids (purple) is computed.

In step 3, the primary assumption is that an object will move between frames, and the distance between the centroids for the current frame F_t and next frame F_{t+1} will be smaller than all other distances between objects. Therefore, the minimum distances between subsequent frames are chosen for association. In step 3, the centroids that minimize the respective Euclidean distances are associated. Thus, the point at the bottom is not associated.

In step 4 (register new objects), if there are more new objects than existing objects being tracked, new objects are registered. They are assigned new object IDs and the centroids of the bounding box coordinates are saved. Step 4 in Figure 6 demonstrates the process of using the minimum Euclidean distances to associate existing object IDs and then registering a new object [38].

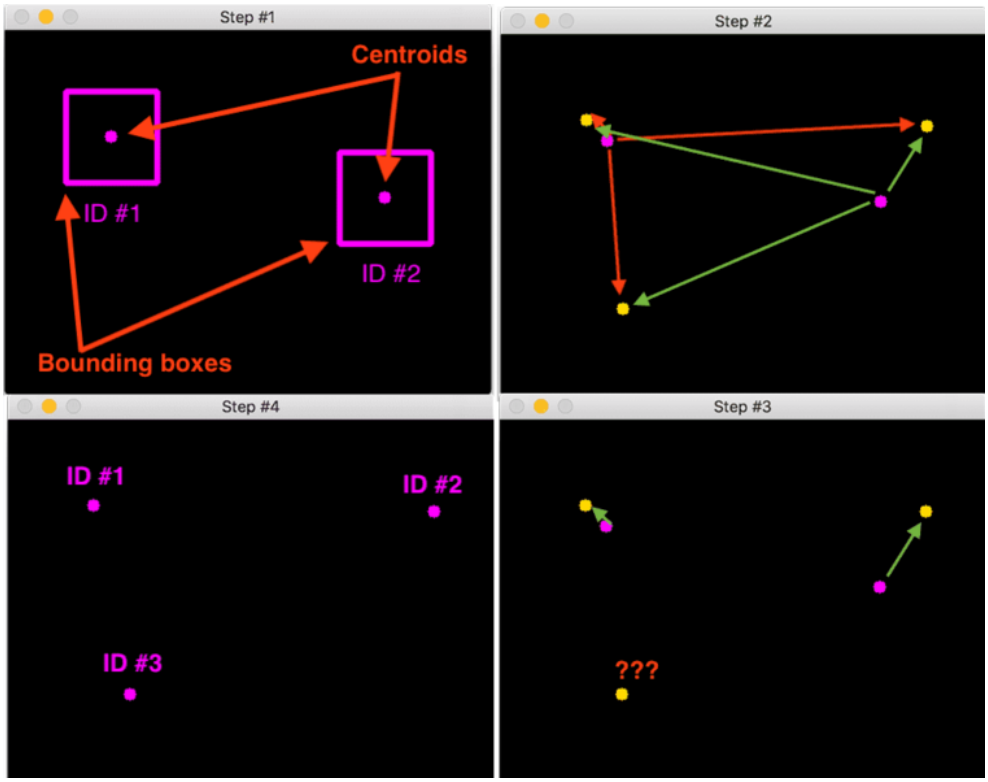


Figure 6: The steps involved in the CRA [38].

3.2. Computation Workflow

The computation workflow for the sensor node is shown in Figure 7. It includes video capture, preprocessing, and traffic parameter extraction tasks implemented in Python using the OpenCV image processing library. The workflow steps are as follows.

1. A Python script is executed to load the pre-trained MobileNet-SSD model onto the compute stick. This is part of the OpenVINO Toolkit which is a deep learning model for detecting objects from an image or video. Compared to YOLO and Faster R-CNN, MobileNet-SSD is better suited for resource constrained devices and provides good real-time object detection accuracy [26]. The MobileNet-SSD model consists of two components, namely the backbone model and SSD head. The backbone model classifies objects using an image classification network for feature extraction, while the SSD head is a convolution layer that creates bounding boxes around the detected objects [27]. The proposed algorithm uses SSD and MobileNet to detect vehicles using a pre-trained Caffe model [28]. The .prototxt prototype Machine Learning (ML) model is loaded onto the compute stick for use with the

Caffe framework [7]. This model has been trained on over 70 objects, but the focus here is on the classification of vehicles as cars, buses, motorbikes, bicycles, and animal drawn carts.

2. The parameters for the CRA are first loaded from a configuration file to the RPi. This is an algorithm in OpenCV used for object tracking in images or video using the Euclidean distance between pixels in consecutive frames [29]. The CRA can be used in combination with object detection models to calculate and store object coordinates. In this work, the MobileNet-SSD model is used to obtain vehicle coordinates. The distance traveled by a vehicle is estimated using the difference between coordinates in successive video frames. The distance traveled is then used to estimate the speed.
3. The roadside video is captured at 30 fps and 1080p resolution using the OpenCV video capture function. The detection model and CRA are used to lower the frame resolution. Although a higher resolution and fps may improve accuracy [30], due to computational constraints, the maximum possible resolution is 360p at 20 fps.
4. The video frames are processed individually. First, a frame is checked to determine if it has already been processed. If not, it is passed through the MobileNet-SSD model for object detection and classification.
5. If a new vehicle is detected in a video frame, it is classified as either a car, bus, motorcycle, bike, or animal drawn cart, and assigned a unique ID. The frame is then passed to the CRA to estimate speed.
6. After object classification and speed estimation, the results are sent to ThingSpeak and simultaneously written to the log file. This process is repeated until a termination command is issued. Upon termination, the log file is transmitted to Dropbox, and execution is stopped.

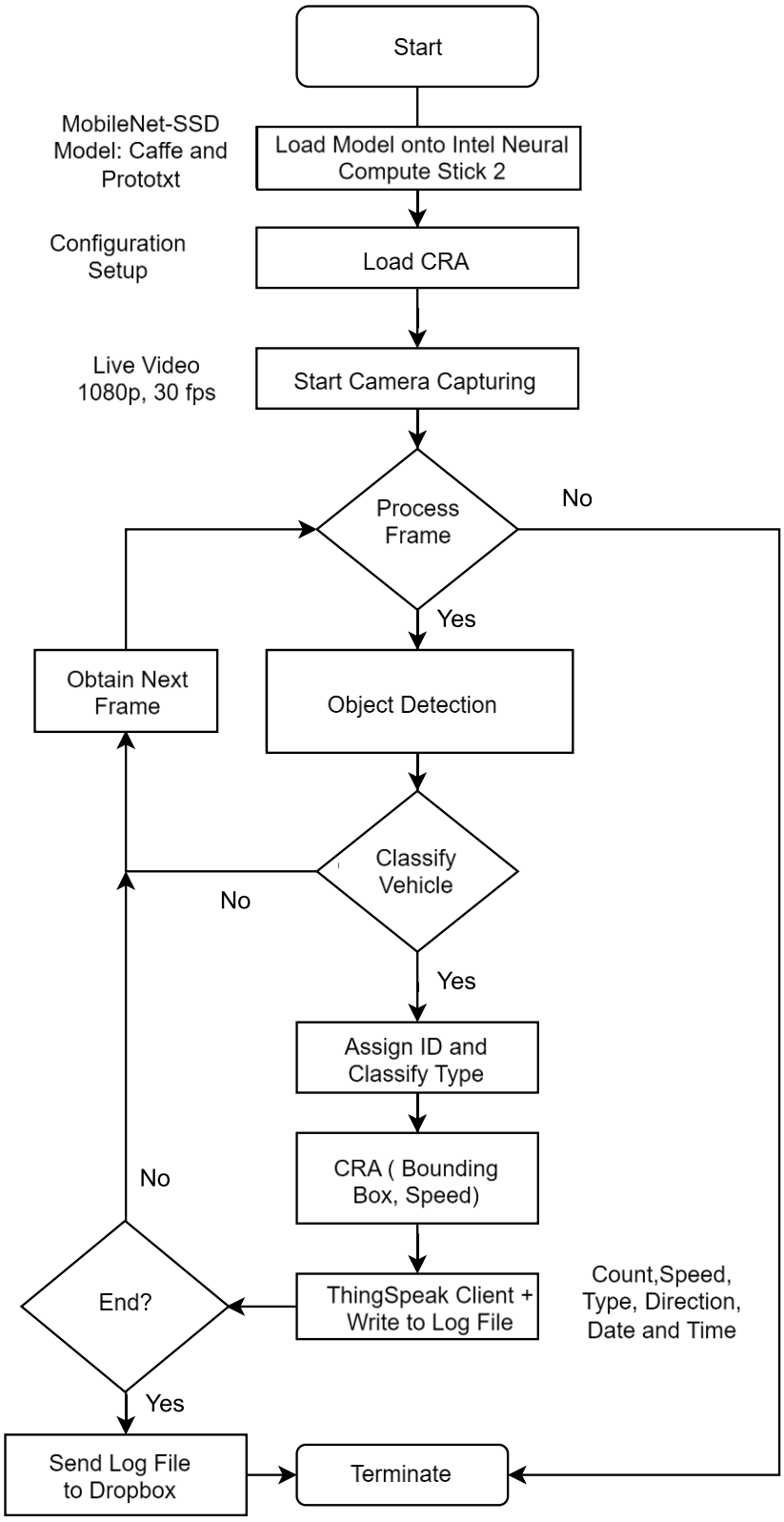


Figure 7: The system computation workflow.

3.3. Cloud Platform

ThingSpeak is a cloud platform that facilitates communication with internet-enabled edge devices through APIs. The platform is free and open source. The ThingSpeak libraries are installed on the sensor node RPi. The ThingSpeak client transmits vehicle count, speed, direction and type, flow, peak hour factor, density, time headway, and distance headway every 15 s. An example of the data for the road under observation is shown in Figure 8.

	A	B	C	D	E	G	H	I	J	K	L	M
1	Date	Time	Speed(in	Direction	Vechicle Type	Average Speed	Traffic Volun	Flow Rate	Peak Hou	Traffic De	Time Hea	Distance Headv
2	10/1/2022	9:08:00 AM	63.2403	Left	motorbike	63.64336478	21	1260	0.552632	0.329964	2.857143	3030.636418
3	10/1/2022	9:08:01 AM	9.457583	Left	car	46.76852811	21	1260	0.552632	0.44902	2.857143	2227.072767
4	10/1/2022	9:08:02 AM	39.21163	Left	car	53.41497646	18	1080	0.473684	0.336984	3.333333	2967.498692
5	10/1/2022	9:08:04 AM	107.4997	Left	car	55.08512958	15	900	0.394737	0.272306	4	3672.341972
6	10/1/2022	9:08:06 AM	58.38977	Left	car	70.35443592	13	780	0.342105	0.184779	4.615385	5411.879686
7	10/1/2022	9:08:07 AM	11.6329	Left	car	60.76984699	20	1200	0.526316	0.329111	3	3038.49235
8	10/1/2022	9:08:07 AM	76.41941	Left	car	59.74355858	16	960	0.421053	0.267811	3.75	3733.972411
9	10/1/2022	9:08:09 AM	96.71847	Left	car	55.61998107	15	900	0.394737	0.269687	4	3707.998738
10	10/1/2022	9:08:11 AM	63.98765	Left	car	57.40758647	18	1080	0.473684	0.313547	3.333333	3189.310359
11	10/1/2022	9:08:12 AM	25.43562	Left	car	65.10848426	16	960	0.421053	0.245744	3.75	4069.280266
12	10/1/2022	9:08:14 AM	118.4443	Left	car	45.12262351	18	1080	0.473684	0.398913	3.333333	2506.812417
13	10/1/2022	9:08:24 AM	68.36612	Left	car	62.78785968	19	1140	0.5	0.302606	3.157895	3304.624194
14	10/1/2022	9:08:26 AM	98.60409	Left	car	55.14065674	22	1320	0.578947	0.39898	2.727273	2506.393488
15	10/1/2022	9:08:29 AM	75.84554	Left	car	53.05136191	13	780	0.342105	0.245046	4.615385	4080.873993
16	10/1/2022	9:08:32 AM	11.13564	Left	car	53.14470708	14	840	0.368421	0.263432	4.285714	3796.050506
17	10/1/2022	9:08:33 AM	62.40659	Left	car	59.61537202	18	1080	0.473684	0.301936	3.333333	3311.965112
18	10/1/2022	9:08:40 AM	95.56835	Left	car	49.42310168	16	960	0.421053	0.323735	3.75	3088.943855
19	10/1/2022	9:08:46 AM	87.97232	Left	car	53.93452441	14	840	0.368421	0.259574	4.285714	3852.466029
20	10/1/2022	9:08:53 AM	76.48511	Left	car	63.97689336	24	1440	0.631579	0.375135	2.5	2665.70389
21	10/1/2022	9:08:55 AM	33.1172	Left	car	62.8484971	16	960	0.421053	0.25458	3.75	3928.031069
22	10/1/2022	9:08:58 AM	39.98088	Left	car	73.69811766	19	1140	0.5	0.257808	3.157895	3878.848298

Figure 8: The real-time traffic parameters from the sensor node for the road under observation.

To ensure data reliability, traffic parameters are stored in a log file in the RPi system root directory called log.csv. Every time a vehicle is detected, a record is added to this file that includes the date (year, month, and day), time, vehicle type, speed, and direction. This serves as a backup of the traffic data that can be referred to in the future if necessary. In addition, at the end of the observation period, the log file is uploaded to Dropbox which is a cloud file sharing and archiving platform. This allows for remote sharing and access of traffic data from Dropbox servers, providing greater flexibility and convenience.

In designing the proposed solution, a key consideration was low operational costs. To achieve this, efforts were made to minimize data bandwidth requirements. Edge computing solutions have a distinct advantage over IoVT-based solutions as they reduce the required bandwidth, resulting in lower costs [13].

To examine data bandwidth requirements for the proposed solution, the sensor node was connected through a Huawei Nova 3i smartphone Wi-Fi hotspot. The bandwidth used by the node was estimated to be 1.5 MB per hour. In Pakistan, a standard monthly 5 GB 4G internet data package costs approximately USD 15, and the proposed solution can transmit traffic data to the cloud platform for approximately 3300 h with this package.

Chapter 4

Results

University Road in Peshawar, Pakistan was chosen to evaluate the system. This is the main arterial road in the city and has few pedestrians. The sensor node was installed south of Islamia College on the north side of the Bus Rapid Transport (BRT) station as shown in Figure 9. This location had poles suitable for the sensor node installation and the correct orientation. The GPS coordinates of the installation location are 33.99841270487691 latitude and 71.47962908395094 longitude. This road connects major institutions such as universities and government organizations, making it the most heavily traversed road in Peshawar. It is a bidirectional road, but the sensor node was installed on the side where traffic runs west to east to prevent direct sunlight on the camera which would compromise the data obtained. The sensor node was positioned perpendicular to the traffic flow as illustrated in Figure 10. It was used to collect data for seven days from Monday, 10 January 2022 to Sunday, 16 January 2022. Each day, the node was in operation for seven hours from 9:00 AM to 4:00 PM. The weather was clear during this time.

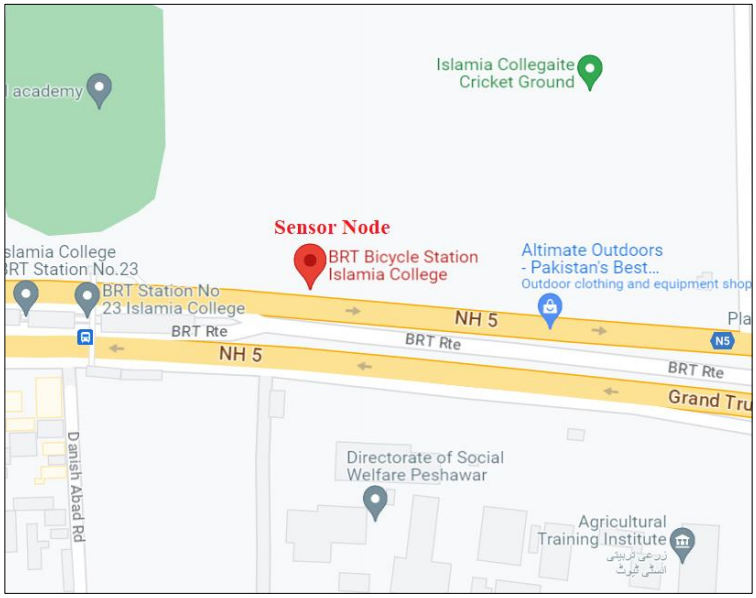


Figure 9: Google Maps location of the sensor node.

Figure 10: The sensor node installation: (a) overhead location, (b) field view, and (c) inner view.

Throughout the evaluation period, the sensor node performed reliably with no issues such as power interruptions or system crashes. It successfully captured traffic data as shown in Table 1. Each day, about 10,000 vehicles traversed the road segment during the seven hours from 9:00 AM to 4:00 PM, and over the week, 69,285 vehicles were observed. The majority of the vehicles were cars (81.0%), followed by motorcycles (15.6%), buses (2.5%), bicycles (0.72%), and animal-drawn carts (0.16%). The highest and lowest traffic volumes were on Monday and Thursday, respectively, as shown in Table 1.

Table 1: Traffic flow on University Road in Peshawar, Pakistan during the week of January 10, 2022 to January 16, 2022.

Day	Cars	Buses	Motorbikes	Bicycles	Animal Drawn Carts	Total
Monday	9821	290	2041	62	23	12,237
Tuesday	7004	220	1078	75	16	8393
Wednesday	8635	244	1989	60	13	10,941
Thursday	7834	73	940	45	12	8904
Friday	7206	283	1604	145	17	9255
Saturday	7608	280	1645	51	18	9602
Sunday	8008	375	1495	61	14	9953
Total	56,116	1765	10,792	499	113	69,285
Percentage	81.0%	2.5%	15.6%	0.72%	0.16%	100%

4.1. Sensor Node Accuracy

Accurate estimation of traffic parameters is critical for an effective monitoring system. In this section, the vehicle count and speed obtained were examined to evaluate the proposed solution. The accuracy was determined for a randomly selected one-hour period from 9:00 AM to 10:00 AM on Saturday, 15 January 2022. The corresponding video was manually analyzed to determine the vehicle count, type, and speed, and these results were compared with those obtained using the sensor node. There were 2356 vehicles in the manual count but only 2196 vehicles with the node. Thus, the system count accuracy was 93.2%, as indicated in Table 2. The mean Average Precision (mAP) is a metric used to evaluate the performance of a model. It involves computing the average precision for each class and then taking the mean of these values across all classes

$$mAP = \frac{1}{n}\sum_{k=1}^{n} AP_k$$

where n is the number of classes and AP_k is the average precision, which is the area under the precision-recall curve for class k

$$AP_k = \sum_n P(n) \times \Delta R(n)$$

where $P(n)$ is the precision at the *n*th retrieved item and $\Delta R(n)$ is the change in recall from the (*n*-1)th to *n*th retrieved item [39].

The mAP of the pre-trained model utilized in this work for object classification is 79.8%. This is higher than other object detection models designed for edge devices [31]. The speed of each vehicle was manually determined by measuring the time it took to travel a given distance. These results were compared with those from the sensor node. Table 2 shows that the speed estimation accuracy was 82.9%. The only other speed accuracy result reported in the literature was 90% [22], but this was for a single vehicle measured between two fixed locations.

Table 2: Proposed system performance.

	Manual	**System**	**Difference**	**Error**	**Accuracy**
Vehicle Count	2356	2196	160	6.8%	93.2%
Average Speed (km/h)	54.9	64.3	9.4	17.1%	82.9%

4.1.1. Regression Modelling

The Greenshields traffic flow model [32] is used to predict the relationship between traffic speed and density on a road. According to this model, an increase in density results in an increase in speed until the road reaches capacity (100% density). At capacity, the speed drops to 0, and the road is considered to be in a jammed state. To verify the accuracy of the sensor node, the relationship between speed and density was modeled using the one hour of data taken from the video captured by the sensor node, and the results are shown in Figure 11. The models employed are exponential, linear, logarithmic, polynomial, and power. The coefficient of determination, commonly known as R-Squared (R^2), is a statistical measure that represents the proportion of the variance in the dependent variable that is predictable from the independent variables. It indicates how effectively the data aligns with the regression model and is defined as

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

where SS_{res} is the sum of squares of residuals (the differences between the observed and predicted values) and SS_{tot} is the total sum of squares (the differences between observed values and the mean of the observed values).

R^2 indicates how much of the variation in the dependent variable can be explained by the variation in the independent variable. A value closer to 1 indicates a better fit of the model to the data, while a value closer to 0 indicates a weaker fit. The results in Figure 11 show that a second order polynomial is the best fit. The corresponding polynomials for the sensor node and manual observations are

$$y1 = 12.32x^2 - 76.14x + 103.90 \quad (2)$$

$$y2 = 37.02x^2 - 73.87x + 84.57 \quad (3)$$

respectively. The error between (2) and (3) is

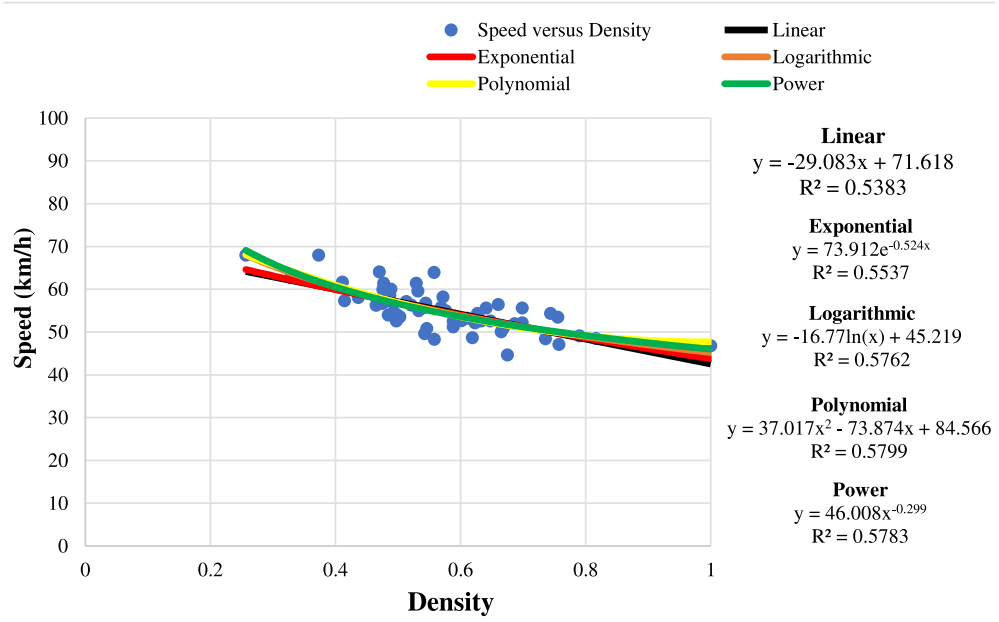
$$\frac{\left|\left|y2\right|\right| - \left|\left|y1\right|\right|}{\left|\left|y2\right|\right|} \quad (4)$$

where

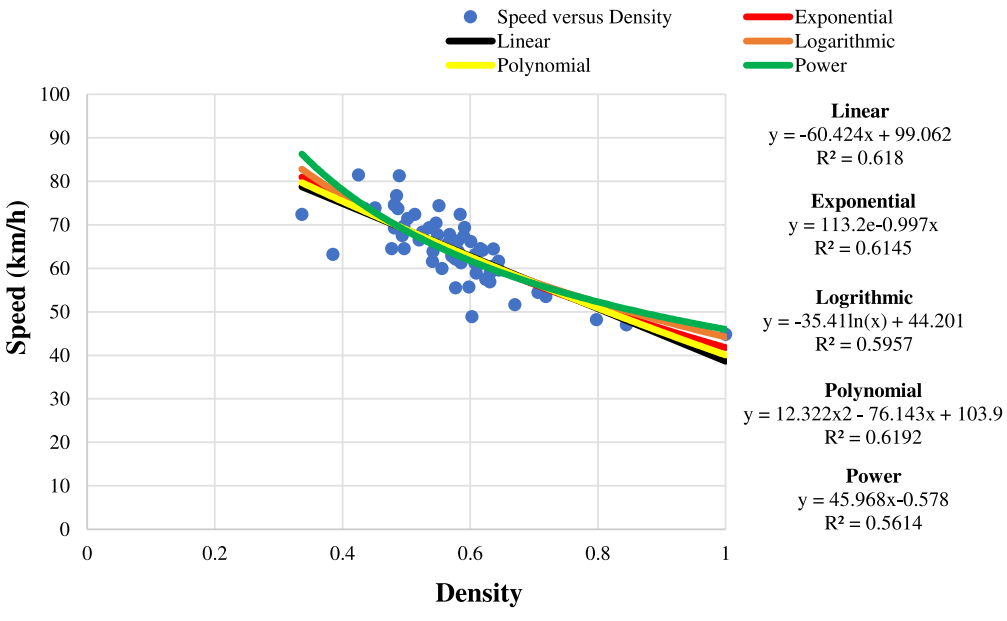
$$\left|\left|y1\right|\right| = \sqrt{y1_1 + y1_2 + y1_3 + \ldots + y1_n} \quad (5)$$

$$\left|\left|y2\right|\right| = \sqrt{y1_1 + y1_2 + y1_3 + \ldots + y1_n} \quad (6)$$

This gives an error of 17.1%.



(a)



(b)

Figure 11: Speed versus density for the one-hour period obtained (a) manually, and (b) from the sensor node.

4.1.2. Limitations and Challenges

The system developed has undergone rigorous testing to evaluate its accuracy and efficiency. This revealed some limitations and challenges which are discussed below.

1. Camera orientation is a significant factor affecting sensor node accuracy. Thus, the camera should be oriented to best capture vehicles passing through the area of interest. It was observed that smaller vehicles are sometimes masked by larger vehicles, resulting in missed detections.
2. Although other ML models may provide more accurate vehicle classification, the proposed solution employs MobileNet-SSD due to its computational efficiency [33]. It was specifically designed for edge computing solutions, and its lightweight computational footprint makes it a good choice for the system.
3. The choice of video resolution and fps can significantly affect the accuracy. However, a higher resolution and fps result in greater computational requirements. Considering this tradeoff, 20 fps is used which resulted in larger errors compared to 25 fps [31], e.g., greater Euclidean distance errors with the CRA [34].
4. The proposed sensor node was tested on a busy road with heterogeneous traffic where vehicles do not follow lane discipline. This behavior will increase the error in calculating speed and other parameters. Therefore, the accuracy of the sensor node results will vary depending on the traffic environment.

Note that despite these limitations and challenges, the proposed solution provides a reliable and efficient means of counting vehicles and estimating their speed.

4.1.3. Relationships Between Density, Flow, and Speed

Table 3 presents the relationships between density, flow, and speed. The equations are based on the data taken by the sensor node.

Table 3: Equations for the relationships between density, flow, and speed for the seven days and the corresponding R^2 values.

Days		Exponential	Linear	Logarithmic	Polynomial	Power
Monday	**Density-Flow**	$y = 24.18e^{0.8005x}$	$y = 37.72x + 16.53$	$y = 31.07\ln(x) + 54.52$	$y = 16.32x^2 + 66.30x + 4.70$	$y = 54.32x^{0.6727}$
		$R^2 = 0.7024$	$R^2 = 0.7301$	$R^2 = 0.7406$	$R^2 = 0.7406$	$R^2 = 0.7379$
	Density-Speed	$y = 81.59e^{-0.405x}$	$y = -23.32x + 78.08$	$y = -19.01\ln(x) + 54.63$	$y = 5.97x^2 - 33.78x + 82.41$	$y = 54.32x^{-0.327}$
		$R^2 = 0.4337$	$R^2 = 0.4317$	$R^2 = 0.4289$	$R^2 = 0.4339$	$R^2 = 0.4226$
	Speed-Flow	$y = 30.46e^{-0.005x}$	$y = -0.12x + 29.78$	$y = -5.94\ln(x) + 47.13$	$y = -0.0098x^2 + 1.03x - 3.19$	$y = 65.34x^{-0.261}$
		$R^2 = 0.0362$	$R^2 = 0.0382$	$R^2 = 0.0309$	$R^2 = 0.0714$	$R^2 = 0.0289$
Tuesday	**Density-Flow**	$y = 15.91e^{1.0242x}$	$y = 29.06x + 13.11$	$y = 17.51\ln(x) + 40.31$	$y = -22.99x^2 + 59.39x + 3.89$	$y = 42.14x^{0.648}$
		$R^2 = 0.4785$	$R^2 = 0.5123$	$R^2 = 0.5399$	$R^2 = 0.5332$	$R^2 = 0.5269$
	Density-Speed	$y = 75.22e^{-0.635x}$	$y = -33.16x + 72.51$	$y = -18.76\ln(x) + 42.07$	$y = 4.49x^2 - 39.09x + 74.32$	$y = 42.14x^{-0.352}$
		$R^2 = 0.3249$	$R^2 = 0.3258$	$R^2 = 0.3028$	$R^2 = 0.3262$	$R^2 = 0.2825$
	Speed-Flow	$y = 24.25e^{0.0045x}$	$y = 0.08x + 27.44$	$y = 6.33\ln(x) + 6.86$	$y = -0.016x^2 + 1.70x - 12.09$	$y = 8.85x^{0.3167}$
		$R^2 = 0.0107$	$R^2 = 0.0139$	$R^2 = 0.0364$	$R^2 = 0.1562$	$R^2 = 0.0279$
Wednesday	**Density-Flow**	$y = 13.96e^{1.427x}$	$y = 36.05x + 11.03$	$y = 15.80\ln(x) + 40.50$	$y = -33.98x^2 + 70.70x + 2.72$	$y = 48.44x^{0.7282}$
		$R^2 = 0.6315$	$R^2 = 0.6807$	$R^2 = 0.6861$	$R^2 = 0.7047$	$R^2 = 0.6950$
	Density-Speed	$y = 85.39e^{-0.743x}$	$y = -44.27x + 81.61$	$y = -16.67\ln(x) + 47.48$	$y = -10.60x^2 - 33.45x + 79.01$	$y = 48.44x^{-0.272}$
		$R^2 = 0.4085$	$R^2 = 0.4148$	$R^2 = 0.3087$	$R^2 = 0.4157$	$R^2 = 0.2316$

	Speed-Flow	$y = 31.19e^{-0.002x}$	$y = -0.08x + 33.90$	$y = -4.67\ln(x) + 47.71$	$y = -0.0064x^2 + 0.69x + 10.57$	$y = 36.84x^{-0.067}$
		$R^2 = 0.0188$	$R^2 = 0.0192$	$R^2 = 0.0153$	$R^2 = 0.0347$	$R^2 = 0.015$
Thursday	**Density-Flow**	$y = 10.44e^{1.6007x}$	$y = 34.45x + 6.90$	$y = 15.30\ln(x) + 35.50$	$y = -28.98x^2 + 63.67x + 0.26$	$y = 40.84x^{0.7534}$
		$R^2 = 0.6215$	$R^2 = 0.6841$	$R^2 = 0.699$	$R^2 = 0.7061$	$R^2 = 0.6956$
	Density-Speed	$y = 66.30e^{-0.606x}$	$y = -29.92x + 64.73$	$y = -12.47\ln(x) + 40.56$	$y = -2.45x^2 - 27.44x + 64.16$	$y = 40.84x^{-0.247}$
		$R^2 = 0.2740$	$R^2 = 0.2758$	$R^2 = 0.2484$	$R^2 = 0.2759$	$R^2 = 0.2333$
	Speed-Flow	$y = 21.11e^{0.0007x}$	$y = -0.0009x + 22.85$	$y = 0.74\ln(x) + 19.92$	$y = -0.0084x^2 + 0.84x + 2.28$	$y = 15.46x^{0.0882}$
		$R^2 = 3E\text{-}06$	$R^2 = 2E\text{-}06$	$R^2 = 0.0004$	$R^2 = 0.0200$	$R^2 = 0.0004$
Friday	**Density-Flow**	$y = 16.116e^{0.9749x}$	$y = 25.69x + 14.32$	$y = 15.50\ln(x) + 38.24$	$y = -28.26x^2 + 63.84x + 2.32$	$y = 42.05x^{0.686}$
		$R^2 = 0.4676$	$R^2 = 0.5145$	$R^2 = 0.5578$	$R^2 = 0.5604$	$R^2 = 0.5363$
	Density-Speed	$y = 74.74e^{-0.667x}$	$y = -32.37x + 70.33$	$y = -16.04\ln(x) + 42.00$	$y = -12.89x^2 - 14.96x + 64.86$	$y = 42.05x^{-0.314}$
		$R^2 = 0.3657$	$R^2 = 0.3777$	$R^2 = 0.2761$	$R^2 = 0.3821$	$R^2 = 0.2021$
	Speed-Flow	$y = 24.13e^{0.0039x}$	$y = 0.035x + 28.36$	$y = 3.04\ln(x) + 18.27$	$y = -0.012x^2 + 1.23x - 0.38$	$y = 10.45x^{0.2648}$
		$R^2 = 0.0020$	$R^2 = 0.0027$	$R^2 = 0.0092$	$R^2 = 0.0649$	$R^2 = 0.0069$
Saturday	**Density-Flow**	$y = 16.12e^{0.9749x}$	$y = 25.69x + 14.31$	$y = 15.50\ln(x) + 38.24$	$y = -28.26x^2 + 63.84x + 2.32$	$y = 42.05x^{0.686}$
		$R^2 = 0.4676$	$R^2 = 0.5145$	$R^2 = 0.5578$	$R^2 = 0.5604$	$R^2 = 0.5363$
	Density-Speed	$y = 74.75e^{-0.667x}$	$y = -32.37x + 70.33$	$y = -16.04\ln(x) + 42.00$	$y = -12.90x^2 - 14.96x + 64.86$	$y = 42.05x^{-0.314}$
		$R^2 = 0.3657$	$R^2 = 0.3777$	$R^2 = 0.2761$	$R^2 = 0.3821$	$R^2 = 0.2021$

	Speed-Flow	$y = 24.14e^{0.0039x}$	$y = 0.035x + 28.36$	$y = 3.04\ln(x) + 18.27$	$y = -0.01x^2 + 1.23x - 0.38$	$y = 10.45x^{0.2648}$
		$R^2 = 0.0020$	$R^2 = 0.0027$	$R^2 = 0.0092$	$R^2 = 0.0649$	$R^2 = 0.0069$
Sunday	**Density-Flow**	$y = 15.64e^{1.2584x}$	$y = 34.05x + 12.85$	$y = 16.18\ln(x) + 41.63$	$y = -44.34x^2 + 80.02x + 1.62$	$y = 46.74x^{0.6402}$
		$R^2 = 0.6192$	$R^2 = 0.6715$	$R^2 = 0.7143$	$R^2 = 0.7159$	$R^2 = 0.6954$
	Density-Speed	$y = 92.46e^{-0.842x}$	$y = -50.28x + 86.43$	$y = -21.77\ln(x) + 45.49$	$y = -6.55x^2 - 43.47x + 84.76$	$y = 46.74x^{-0.36}$
		$R^2 = 0.5347$	$R^2 = 0.5416$	$R^2 = 0.4784$	$R^2 = 0.5420$	$R^2 = 0.4203$
	Speed-Flow	$y = 40.68e^{-0.005x}$	$y = -0.16x + 39.33$	$y = -8.61\ln(x) + 65.18$	$y = -0.0065x^2 + 0.64x + 15.14$	$y = 99.11x^{-0.297}$
		$R^2 = 0.0624$	$R^2 = 0.0648$	$R^2 = 0.0561$	$R^2 = 0.0837$	$R^2 = 0.0531$

Table 3 shows that the polynomial equations have better R^2 values so these are chosen for comparison. The highest R^2 values are on Monday ($R^2 = 0.7406$) and Sunday ($R^2 = 0.7159$), indicating that the polynomial equations for these days provide the best fit for the density-flow relationship. For density-speed, the highest R^2 value was observed on Sunday ($R^2 = 0.5420$), indicating the best fit. For speed-flow, the highest R^2 value was observed on Tuesday ($R^2 = 0.1562$).

4.2. Traffic Parameters

The proposed solution utilizes CV algorithms to obtain vehicle count, type, direction, and speed. The remaining traffic parameters, namely, flow, peak hour factor, density, time headway, and distance headway are obtained as follows. Flow is the number of vehicles that pass a point on a road per unit of time and is given by

$$q = \frac{n}{t} \tag{7}$$

where n is the number of vehicles in time t. The peak hour factor is a measure of the traffic flow during a particular hour q_{60} relative to the highest flow in a 15 min period q_{15} on a road and is given by

$$PHF = \frac{q_{60}}{q_{15}} \tag{8}$$

Traffic density is the number of vehicles per unit length of the road and is given by

$$k = \frac{q}{v} \tag{9}$$

Time headway is the distance between consecutive vehicles and can be expressed as

$$h = \frac{1}{q} \tag{10}$$

Distance headway is the distance between consecutive vehicles and is given by

$$s = \frac{1}{k} \tag{11}$$

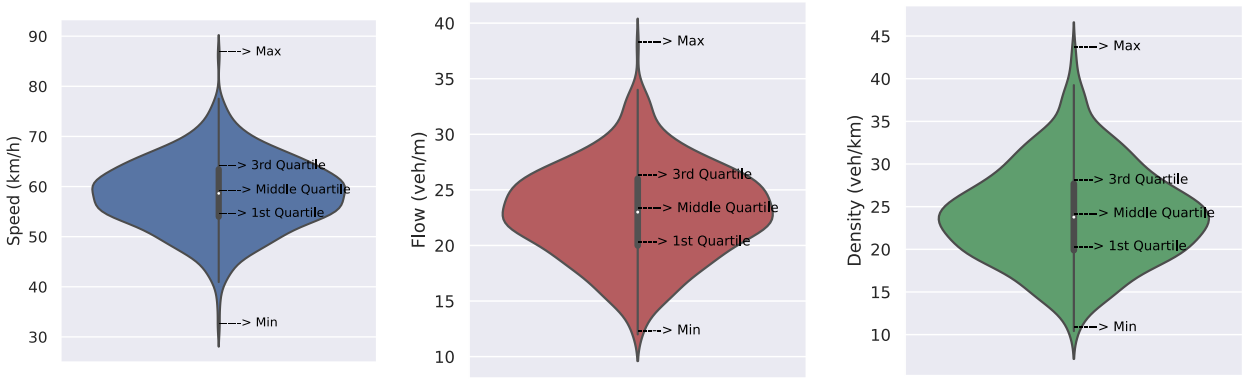
The speed, flow, and density for each of the seven days are given in Table 4. These results are for one-minute intervals. The minimum speed is between 6.1 km/h and 35.7 km/h, the maximum speed between 74.1 km/h and 89.7 km/h, and the mean speed between 50.3 km/h and 61.3 km/h. The minimum flow is between 3.0 veh/m and 12.0 veh/m, the maximum flow is between 32.0 veh/m and 39.0 veh/m, and the mean flow is between 16.6 veh/m and 24.1 veh/m. The standard deviation of the flow is between 3.8 veh/m and 4.9 veh/m, indicating the differences in flow each day. The density ranges from a minimum of 3.9 veh/km to a maximum of 56.9 veh/km. The mean density is between 20.2 veh/km and 27.6 veh/km, and the standard deviation is between 5.5 veh/km and 7.4 veh/km, so the variations each day are similar.

Table 4: Speed (km/h), flow (veh/m), and density (veh/km) statistics for the seven days.

Day	Parameter	Min	First Quartile	Second Quartile	Third Quartile	Max	Mean	Standard Deviation
Monday	**Speed**	32.1	53.9	58.6	63.4	86.1	58.4	7.0
	Flow	12.0	20.0	23.0	24.0	38.0	23.0	4.0
	Density	10.4	19.9	23.7	27.7	43.3	24.0	5.7
Tuesday	**Speed**	15.8	44.6	51.3	58.0	83.8	51.3	10.6
	Flow	4.0	18.0	21.0	24.0	36.0	21.0	4.9
	Density	6.8	20.5	24.9	29.8	48.6	25.4	7.3
Wednesday	**Speed**	35.7	54.9	60.4	66.0	88.7	60.0	8.3
	Flow	12.0	20.0	23.0	26.0	32.0	23.1	4.1
	Density	10.1	19.0	22.5	26.5	48.3	23.6	5.8
Thursday	**Speed**	24.1	44.6	50.8	56.6	74.1	50.9	8.7
	Flow	4.3	20.0	22.0	25.0	34.0	16.6	4.6
	Density	4.3	22.7	26.7	31.5	43.8	20.2	6.7
Friday	**Speed**	18.1	53.1	57.8	63.2	82.1	50.3	8.8
	Flow	5.0	20.0	23.0	25.0	39.0	22.5	4.3
	Density	10.5	20.5	23.7	27.8	56.9	27.6	7.4
Saturday	**Speed**	6.1	53.1	57.8	63.2	83.1	58.2	8.1
	Flow	3.0	20.0	23.0	25.0	35.0	22.7	3.8
	Density	10.3	20.5	23.7	27.8	44.9	23.9	5.5
Sunday	**Speed**	33.6	55.9	61.0	67.1	89.7	61.3	8.4
	Flow	4.0	22.0	24.0	27.0	35.0	24.1	4.2
	Density	3.9	20.0	23.9	27.9	48.5	24.1	6.0

Figure 12 presents violin plots for the speed, flow, and density results. They combine the benefits of box plots and kernel density plots and allow for easy interpretation of the data distributions by quartile regions. Figure 12a gives the speed for Monday, 10 January 2022. The plot is wide in the middle which indicates that the speed of most vehicles is concentrated around 58.4 km/h (mean),

between 53.9 km/h (first quartile) and 63.4 km/h (third quartile). The plot is narrow at the extreme points, 32.1 km/h (minimum) and 86.1 km/h (maximum), so there are very few vehicles at these speeds. Figures 12b,c give the corresponding flow and density plots. They indicate that the flow and density of most vehicles are between 20.0 veh/m and 24.0 veh/m and 19.9 veh/km and 27.7 veh/km, respectively. Similar results were obtained for Tuesday, 11 January 2022 to Friday, 14 January 2022, as shown in Figures 12d–u. For example, the speed is between 44.6 km/h and 67.1 km/h for most vehicles as the plots are wide in the middle. Figures 12p–u give the plots for the weekend (Saturday and Sunday). These plots are smaller than on the weekdays which indicates lower traffic volumes. This is reflected in the higher speeds, mostly between 53.3 km/h and 67.1 km/h. The means are also higher, so traffic conditions are better on these days. Further, the maximum speed of 89.7 km/h occurred on Sunday.

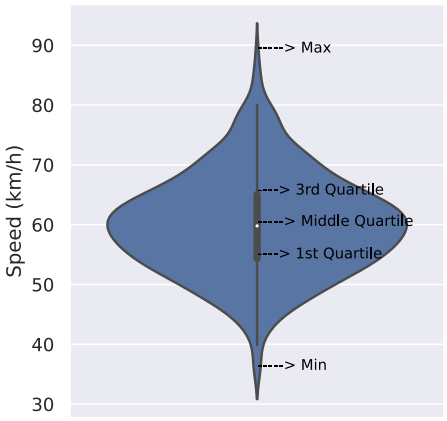


(a) **(b)** **(c)**

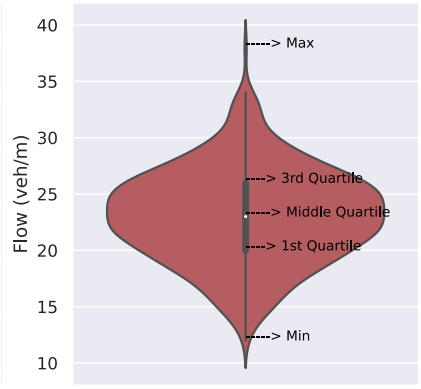
Monday

(d) **(e)** **(f)**

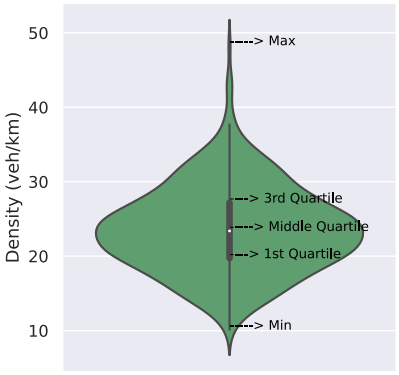
Tuesday



(g)

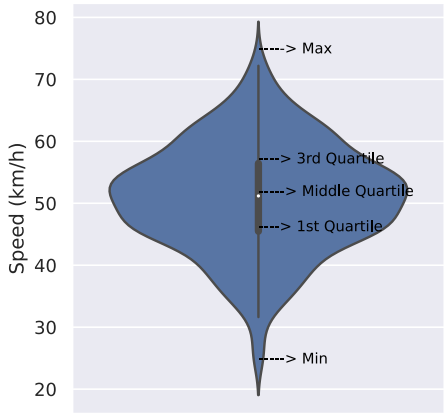


(h)

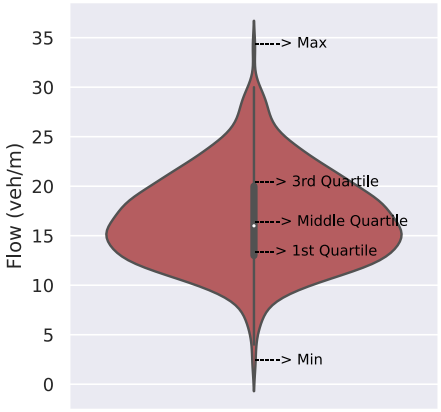


(i)

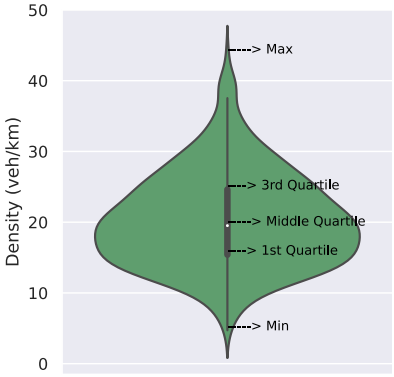
Wednesday



(j)

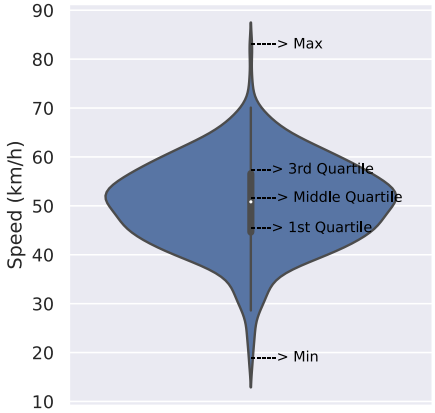


(k)

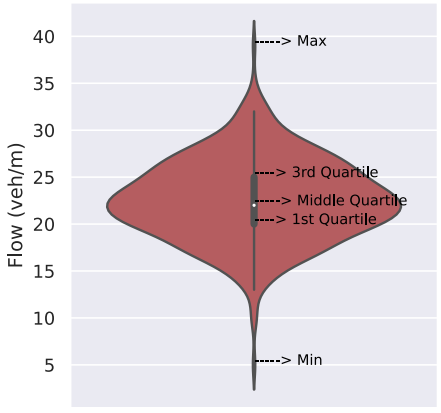


(l)

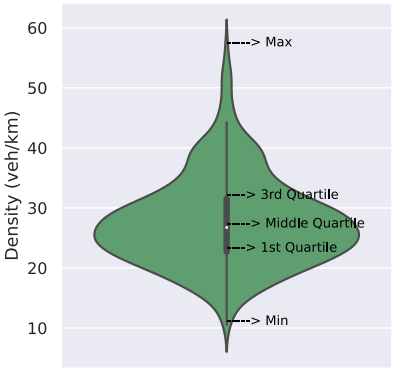
Thursday



(m)



(n)



(o)

Friday

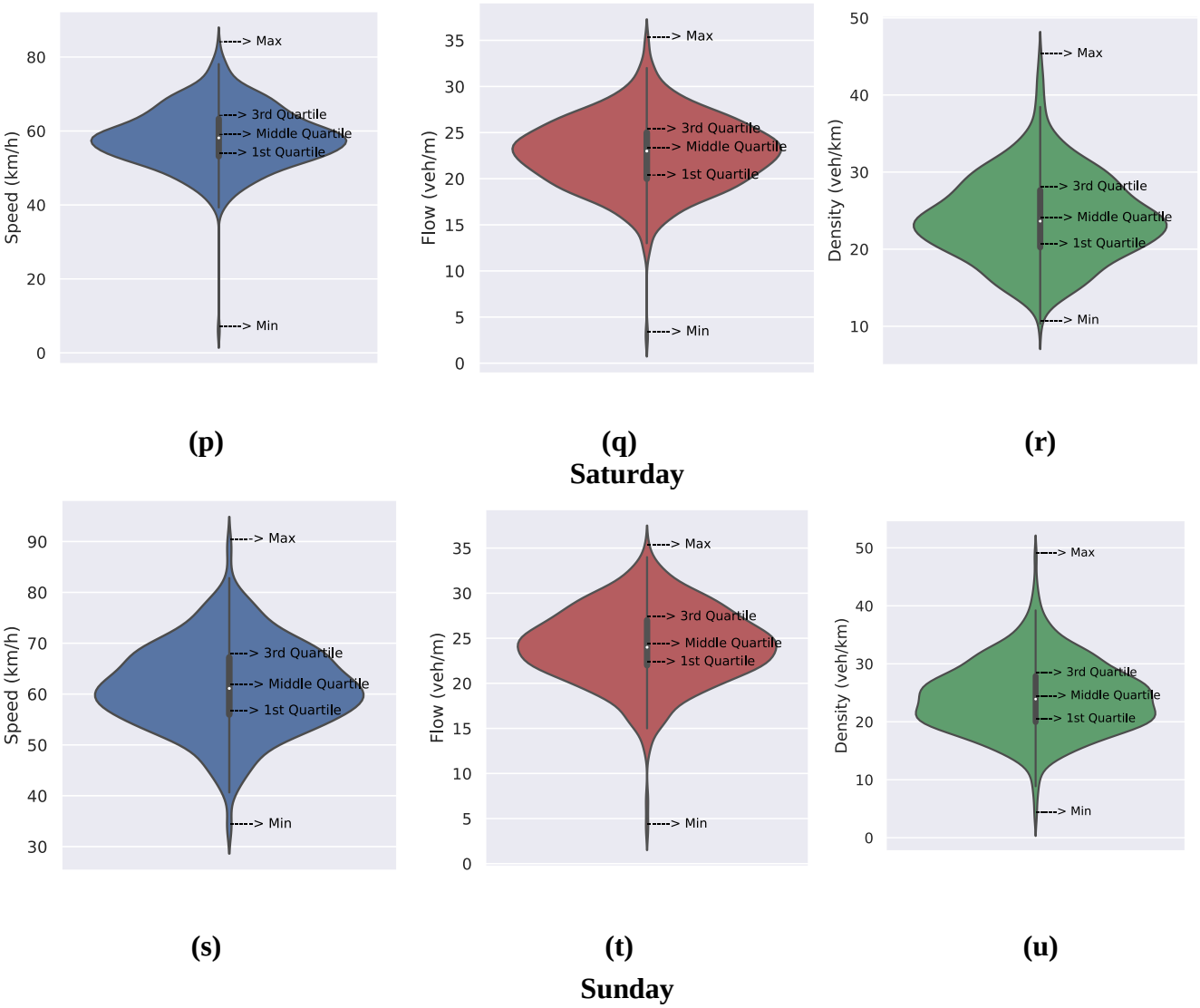


Figure 12: Interquartile representation of speed (blue), flow (red), and density (green) using violin plots for the week of January 10, 2022 – January 16, 2022. (a-c) Monday, 10 January 2022; (d-f) Tuesday, 11 January 2022; (g-i) Wednesday, 12 January 2022; (j-l) Thursday, 13 January 2022; (m-o) Friday, 14 January 2022; (p-r) Saturday, 15 January 2022; and (s-u) Sunday, 16 January 2022.

4.3. Traffic Behavior Analysis

The Greenshields model is widely used to characterize uninterrupted traffic flow conditions [32]. It defines the relationship between speed, density, and flow. In this model, the speed is given by

$$v(k) = vf\left(1 - \frac{k}{k_j}\right) \tag{12}$$

where *vf* is the free flow speed when the density is zero, *k* is the density, and k_j is the jam density (when the speed is zero). The flow can be expressed as

$$q = kv \tag{13}$$

Substituting (12) in (13) gives the relationship between flow and density as

$$q(k) = vf\left(k - \frac{k^2}{k_j}\right) \tag{14}$$

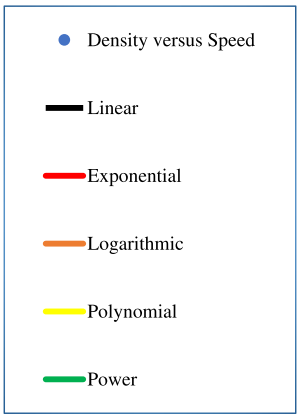
The relationship between flow and speed is then

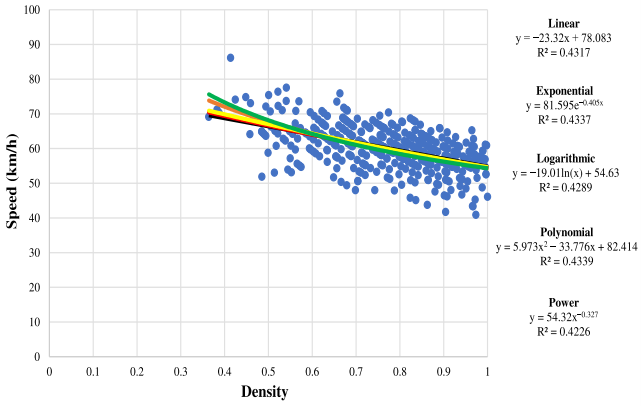
$$q(v) = k_j v - \frac{v^2}{vf} \tag{15}$$

Parameters for vehicles on the road segment were obtained for one-minute intervals, e.g., veh/min. These data were used to model the relationships between density and speed, density and flow, and speed and flow using exponential, linear, logarithmic, polynomial, and power expressions. The corresponding R^2 values indicate that the linear model is the best fit.

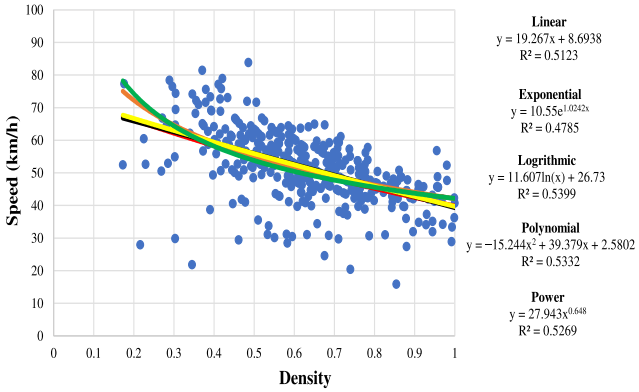
4.3.1. Density Versus Speed

The relationships between density and speed for the seven days are given in Figure 13. These results suggest that the flow on the road is good throughout the week, with the average speed never low even though the density reaches its maximum. Since the road has a high capacity to handle normal traffic, it is likely to be frequently used. Figure 13 shows that the speed is not 0 when the density approaches 1, indicating that even with a high volume, traffic is smooth with no congestion. In addition, the speed does not decrease linearly to 0 with an increase in density beyond 0.5, which differs from the Greenshields traffic model given in (12).

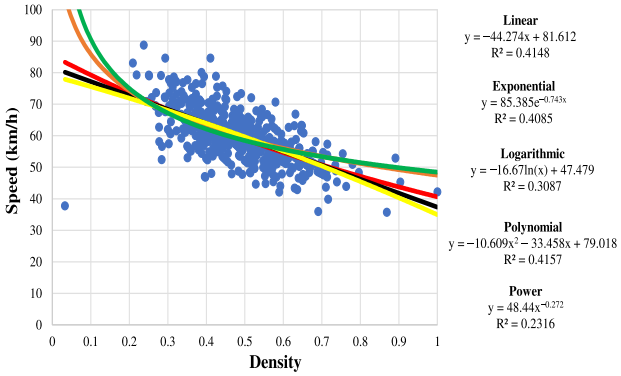

Density versus Speed
Linear
Exponential
Logarithmic
Polynomial
Power


Speed (km/h)
Density
Linear
y = −23.32x + 78.083
R² = 0.4317
Exponential
y = 81.595e^-0.405x
R² = 0.4337
Logarithmic
y = −19.01ln(x) + 54.63
R² = 0.4289
Polynomial
y = 5.973x² − 33.776x + 82.414
R² = 0.4339
Power
y = 54.32x^-0.327
R² = 0.4226

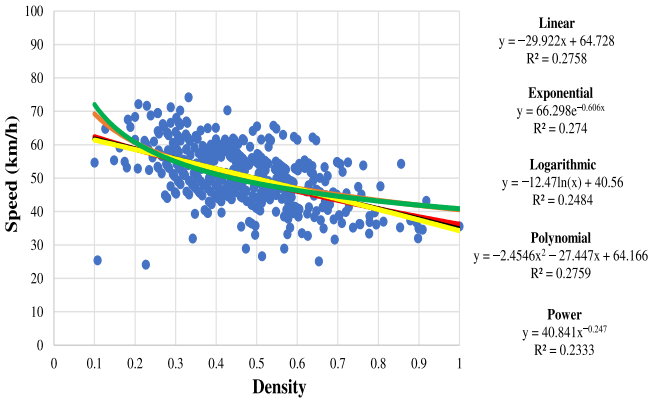
(**a**)


Speed (km/h)
Density
Linear
y = 19.267x + 8.6938
R² = 0.5123
Exponential
y = 10.55e^1.0242x
R² = 0.4785
Logrithmic
y = 11.607ln(x) + 26.73
R² = 0.5399
Polynomial
y = −15.244x² + 39.379x + 2.5802
R² = 0.5332
Power
y = 27.943x^0.648
R² = 0.5269

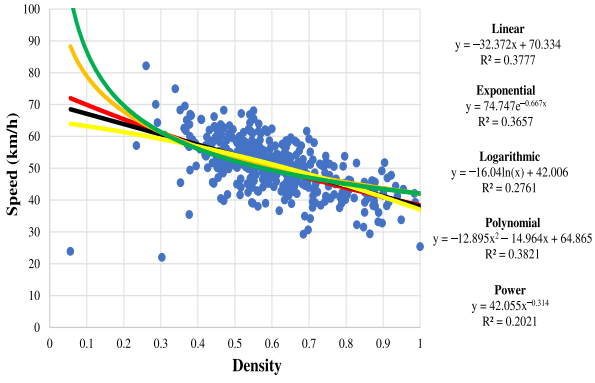
(**b**)


Speed (km/h)
Density
Linear
y = −44.274x + 81.612
R² = 0.4148
Exponential
y = 85.385e^-0.743x
R² = 0.4085
Logarithmic
y = −16.67ln(x) + 47.479
R² = 0.3087
Polynomial
y = −10.609x² − 33.458x + 79.018
R² = 0.4157
Power
y = 48.44x^-0.272
R² = 0.2316

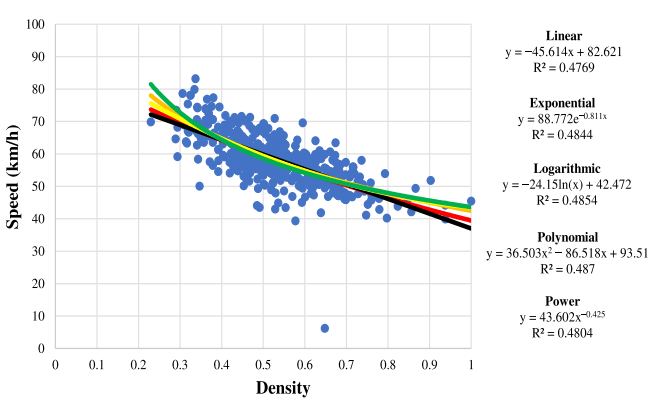
(**c**)


Speed (km/h)
Density
Linear
y = −29.922x + 64.728
R² = 0.2758
Exponential
y = 66.298e^-0.606x
R² = 0.274
Logarithmic
y = −12.47ln(x) + 40.56
R² = 0.2484
Polynomial
y = −2.4546x² − 27.447x + 64.166
R² = 0.2759
Power
y = 40.841x^-0.247
R² = 0.2333

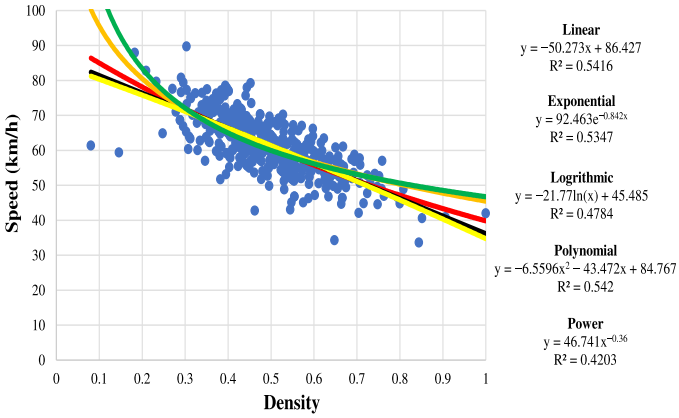
(**d**)


Speed (km/h)
Density
Linear
y = −32.372x + 70.334
R² = 0.3777
Exponential
y = 74.747e^-0.667x
R² = 0.3657
Logarithmic
y = −16.04ln(x) + 42.006
R² = 0.2761
Polynomial
y = −12.895x² − 14.964x + 64.865
R² = 0.3821
Power
y = 42.055x^-0.314
R² = 0.2021

(**e**)



(**f**)

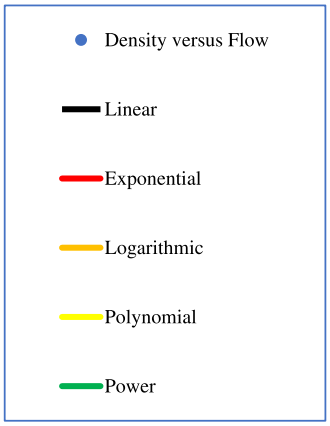


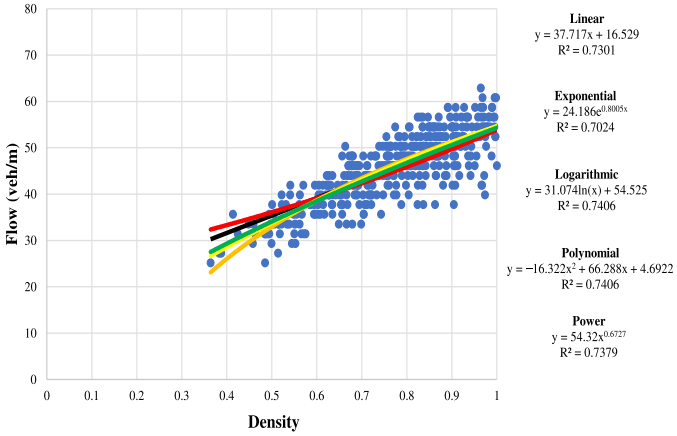
(**g**)

Figure 13: Speed versus density for the week of January 10 to January 16, 2022. (a) Monday, 10 January 2022; (b) Tuesday, 11 January 2022; (c) Wednesday, 12 January 2022; (d) Thursday, 13 January 2022; (e) Friday, 14 January 2022; (f) Saturday, 15 January 2022; and (g) Sunday, 16 January 2022.

4.3.2. Density Versus Flow

Figure 14 gives the relationships between density and flow for the seven days. From (14), the flow increases with density until the road capacity is reached. Once the capacity is exceeded, the flow decreases with a further increase in density. Figure 14 shows that the maximum density occurs only on Monday (a) and Tuesday (b) with a slightly greater flow than on other days, indicating these are the busiest days. Therefore, the road capacity, which for this road segment should range from 60 – 80 veh/m has not been exceeded and does not significantly affect the traffic as there is no congestion.





(**a**)

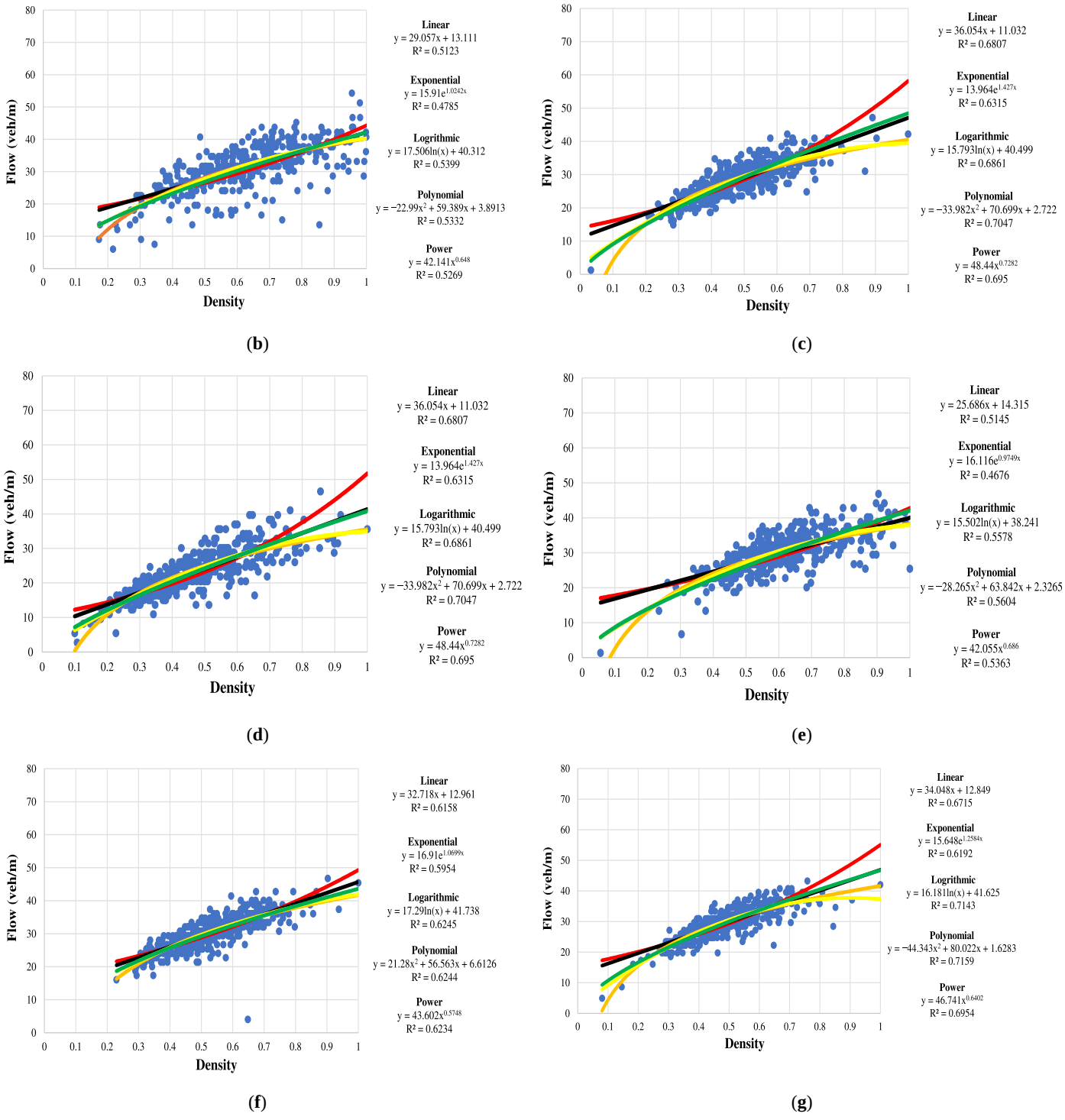
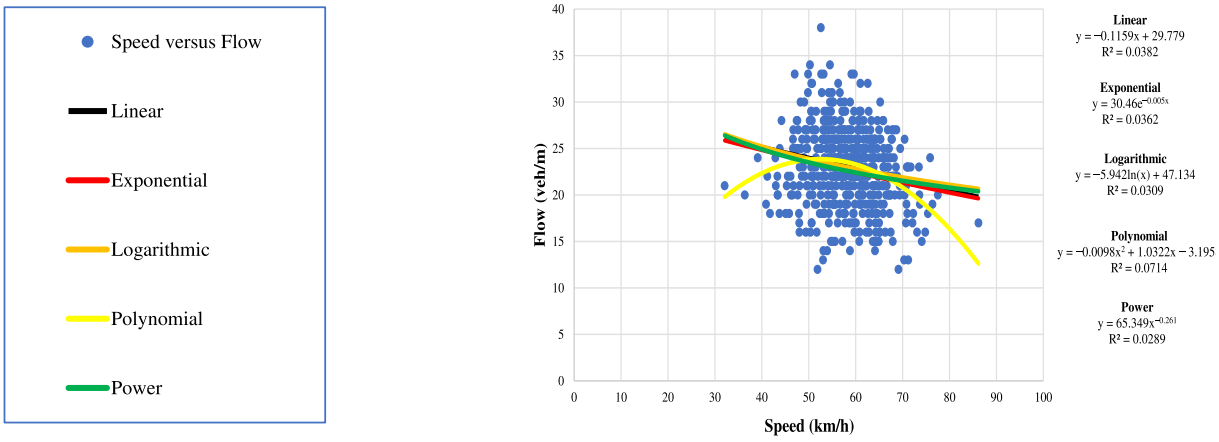


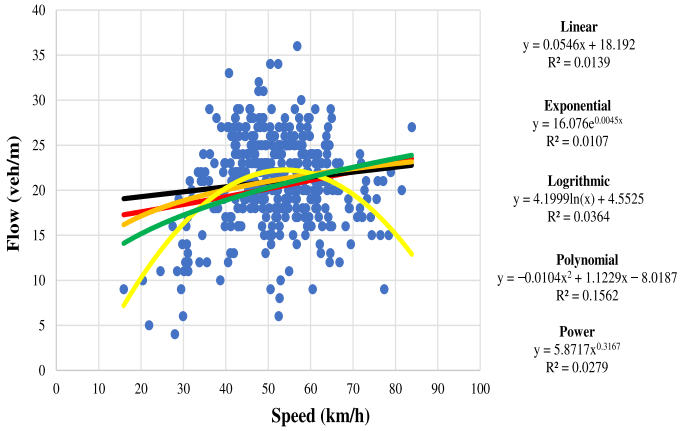
Figure 14: Density versus flow for the week of January 10 to January 16, 2022. (a) Monday, 10 January 2022; (b) Tuesday, 11 January 2022; (c) Wednesday, 12 January 2022; (d) Thursday, 13 January 2022; (e) Friday, 14 January 2022; (f) Saturday, 15 January 2022; and (g) Sunday, 16 January 2022.

4.3.3. Speed Versus Flow

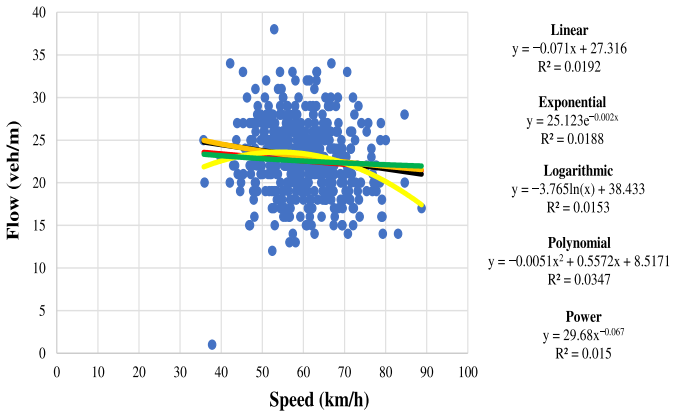
The Greenshields model (15) provides the relationship between speed and flow. According to this model, when there are no vehicles on the road, i.e., zero density, the flow is also zero. As density increases, the flow also increases until it reaches its maximum at the road capacity. Beyond this point, as the density continues to increase, the flow decreases and eventually drops to zero when the density reaches its maximum. The corresponding speed is also zero. Figure 15 gives the relationships between flow and speed for the seven days. This shows that the flow is near maximum and the speed is never low. Thus, the road has a high capacity and is free of congestion. If congestion were to occur, the jam density would be reached, causing the flow and speed to drop to 0.



(**a**)



(**b**)



(**c**)

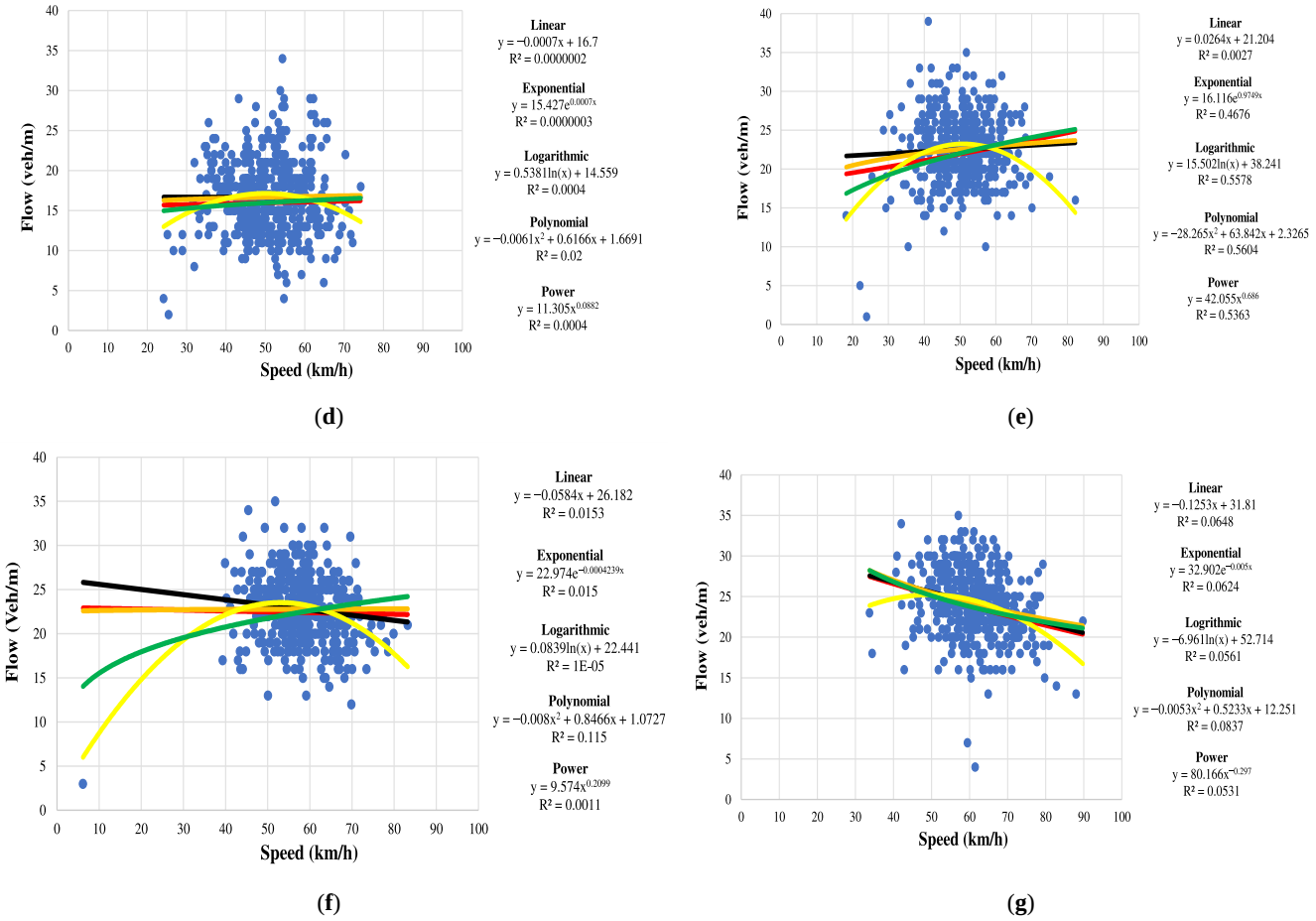


Figure 15: Speed versus flow for the week of January 10 to January 16, 2022. (a) Monday, 10 January 2022; (b) Tuesday, 11 January 2022; (c) Wednesday, 12 January 2022; (d) Thursday, 13 January 2022; (e) Friday, 14 January 2022; (f) Saturday, 15 January 2022; and (g) Sunday, 16 January 2022.

Chapter 5

Conclusion

Intelligent transportation systems encompass a diversity of methodologies for monitoring and aggregating traffic data. They often leverage a variety of devices including pressure sensors, piezoelectric sensors, radar systems, and pneumatic tubes. Developing robust sensor nodes is challenging due to cost, implementation complexity, limited parameter availability, and issues related to data transmission and management. These were overcome here using advanced technologies such as image processing, ML, and cloud storage. The proposed edge computing solution is low cost and energy efficient. It includes an RPi 4, Pi Camera, NCS2, Xiaomi MI Power Bank, and Zong 4G Bolt+. The sensor node is compact so it can easily be installed on a roadside. A key component is the MobileNet-SSD model which is used for accurate vehicle detection. The CRA is also used to estimate velocity. The computational complexity is low, resulting in excellent energy efficiency. As a result, the proposed sensor node is superior to existing solutions.

The system was field evaluated over 7 days for 7 hours a day on a diverse range of traffic. It was able to extract vehicle count, speed, direction, and type, flow, peak hour factor, density, time headway, and distance headway for approximately 10,000 vehicles per day. During this period, there was no congestion, and the flow was smooth with high speeds. The count and speed accuracy were 93.2% and 82.9%, respectively. The power consumption of the sensor node was only 1.2 A per hour.

The Greenshields model was used to characterize the relationships between density, speed, and flow. It was shown that the flow increases with density until the road capacity is reached. The speed versus flow relationship also followed this model. The results suggest that the observed road segment has been well designed and is capable of handling high traffic volumes without creating congestion. The parameters obtained can be used to develop models for use in traffic simulation

software. The proposed edge computing solution can provide valuable insights into road traffic behavior to facilitate intelligent transportation systems and smart urban mobility development.

5.1. Future Work

There are several avenues for future research. Node operation can be extended by incorporating energy harvesting, solar panels, and/or advanced batteries. Energy harvesting can lengthen operation time and enable the node to be used remotely in more applications. The pre-trained model employed provides excellent performance, but other models that can be trained on datasets for particular road segments can be considered. This can improve accuracy, particularly in heterogeneous traffic environments. Moreover, a network of sensor nodes can be deployed to provide insights into traffic dynamics across multiple road segments and at intersections.

References

[1] A. Khan, K. Khattak, Z. H. Khan, M. A. Khan, and N. Minallah, “Cyber Physical System for Vehicle Counting and Emission Monitoring,” *International Journal of Advanced Computer Research*, vol. 10, no. 50, pp. 181–193, Sep. 2020.

[2] Texas A&M Transportation Institute, “2021 Urban Mobility Report,” https://mobility.tamu.edu/umr/. Accessed: 2023-06-22.

[3] World Health Organization, *“*Occupant Restraints: A Road Safety Manual for Decision-Makers and Practitioners,” https://www.who.int/publications/m/item/occupant-restraints--a-road-safety-manual-for-decision-makers-and-practitioners. Accessed: 2023-06-21.

[4] C. Bowman and J. Miller, “Modeling Traffic Flow using Simulation and Big Data Analytics,” in *Proceedings of the Winter Simulation Conference,* Washington, DC, USA, pp. 1206–1217, Dec. 2016.

[5] A. Zeb, K. S. Khattak, M. Rehmat Ullah, Z. H. Khan, and T. A. Gulliver, “HetroTraffSim: A Macroscopic Heterogeneous Traffic Flow Simulator for Road Bottlenecks,” *Future Transportation*, vol. 3, no. 1, pp. 368–383, Mar. 2023.

[6] S. Ayaz, K. S. Khattak, Z. H. Khan, N. Minallah, M. A. Khan, and A.N. Khan, “Sensing Technologies for Traffic Flow Characterization: From Heterogeneous Traffic Perspective,” *Journal of Applied Engineering Science*, vol. 20, no. 1, pp. 29-40, Mar. 2022.

[7] Google, “.PROTOTXT File Extension,” https://fileinfo.com/extension/prototxt. Accessed: 2024-03-03.

[8] A. Gupta, C. Gandhi, V. Katara, and S. Brar, “Real-time Video Monitoring of Vehicular Traffic and Adaptive Signal Change using Raspberry Pi,” in *Proceedings of the IEEE Students’ Conference on Engineering & Systems*, Prayagraj, India, Jul. 2020.

[9] S. Prabhu M, S.R. Kawale, T. Chockalingam, M. Mallampati, D.N. Sahu, and M.A. Huluka, “Automated Vehicle Detection and Tracking using Raspberry-Pi,” *International Journal of Mechanical Engineering*, vol. 7, no. 4, pp. 66–72, Apr. 2022.

[10] K. L.-M. Ang, J. K. P. Seng, E. Ngharamike, and G. K. Ijemaru, “Emerging Technologies for Smart Cities’ Transportation: Geo-Information, Data Analytics and Machine Learning Approaches,” *ISPRS International Journal of Geo-Information*, vol. 11, no. 2, art. 85, Jan. 2022.

[11] A. Zakhil, A. Khan, K.S. Khattak, Z. H. Khan, and A. Qazi, “Vehicular Flow Characterization: An Internet of Video Things Based Solution,” *Pakistan Journal of. Engineering and Technology,* vol. 6, pp. 13–22, Mar. 2023.

[12] A. Jiménez, V. Garcia-Díaz, and J. Anzola, “Design of a System for Vehicle Traffic Estimation for Applications on IoT,” in *Proceedings of the Multidisciplinary International Social Networks Conference*, Bangkok, Thailand, art. 15, Jul. 2017.

[13] C.C. Paglinawan, A.N. Yumang, L.C.M. Andrada, E.C. Garcia, and J.M.F. Hernandez, “Optimization of Vehicle Speed Calculation on Raspberry Pi Using Sparse Random Projection,” in *Proceedings of the IEEE International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management,* Baguio City, Philippines, Nov. 2018.

[14] A. Kulkarni and V. Baligar, “Real Time Vehicle Detection, Tracking and Counting Using Raspberry-Pi,” in *Proceedings of the International Conference on Innovative Mechanisms for Industry Applications*, Bangalore, India, pp. 603–607, Mar. 2020.

[15] A. Suryatali and V. B. Dharmadhikari, “Computer Vision-Based Vehicle Detection for Toll Collection System using Embedded Linux,” in *Proceedings of the International Conference on Circuits, Power and Computing Technologies*, Nagercoil, India, Mar. 2015.

[16] F. T. Espinoza, B. G. Gabriel, and M. J. Barros, “Computer Vision Classifier and Platform for Automatic Counting: More than Cars,” in *Proceedings of the IEEE Ecuador Technical Chapters Meeting*, Salinas, Ecuador, Oct. 2017.

[17] D. Gregor, K. Cikel, M. Arzamendia, and R. Gregor, “Design and Implementation of a Counting and Differentiation System for Vehicles through Video Processing,” *International Journal of Computer and Information Engineering*, vol. 10, no. 10, pp. 1771–1778, Aug. 2016.

[18] T. Sorwar, S. Azad, S. Hussain, and A. Mahmood, “Real-time Vehicle Monitoring for Traffic Surveillance and Adaptive Change Detection using Raspberry Pi Camera Module,” in

Proceedings of the IEEE Region 10 Humanitarian Technology Conference, Dhaka, Bangladesh, pp. 481–484, Dec. 2017.

[19] Open Visual Inference and Neural Network Optimization (OpenVINO), "MobileNet-SSD," https://docs.openvino.ai/2023.0/omz_models_model_mobilenet_ssd.html. Accessed: 2024-03-05.

[20] I. Iszaidy, A. Alias, R. Ngadiran, R. B. Ahmad, M. I. Jais, and D. Shuhaizar, "Video Size Comparison for Embedded Vehicle Speed Detection & Travel Time Estimation System by using Raspberry Pi," in *Proceeding of the International Conference on Robotics, Automation and Sciences,* Melaka, Malaysia, Nov. 2016.

[21] R. McQueen, Directed Studies Report, "Detection and Speed Estimation of Vehicles Using Resource Constrained Embedded Devices," University of British Columbia, Kelowna, BC, Canada, 2018.

[22] R. Ullah, K. Khattak, Z. H. Khan, M. A. Khan, N. Minallah, and A. Khan, "Vehicular Traffic Simulation Software: A Systematic Comparative Analysis," *Pakistan Journal Engineering Technology.* vol. 4, pp. 66–78. Mar. 2021.

[23] G. Tsagkatakis and A. Savakis, "Random Projections for face detection under resource constraints," in *Proceedings of the IEEE International Conference on Image Processing*, Cairo, Egypt, pp. 1233–1236, Dec. 2009.

[24] Y.-C. Chiu, C.-Y. Tsai, M.-D. Ruan, G.-Y. Shen, and T.-T. Lee, "Mobilenet-SSDv2: An Improved Object Detection Model for Embedded Systems," in *Proceedings of the International Conference on System Science and Engineering*, Kagawa, Japan, Aug.-Sep. 2020.

[25] M. Qin, L. Chen, N. Zhao, Y. Chen, F. Yu, and G. Wei, "Power-Constrained Edge Computing with Maximum Processing Capacity for IoT Networks," *IEEE Internet of Things Journal*, vol. 6, pp. 4330-4343, Jun. 2019.

[26] A. Rosebrock, "Object Detection with Deep Learning and OpenCV," https://www.pyimagesearch.com/2017/09/11/object-detection-with-deep-learning-and-opencv/. Accessed: 2024-03-05.

[27] W. Liu et al., "SSD: Single Shot MultiBox Detector," in *Computer Vision-ECCV 2016,* Lecture Notes in Computer Science, vol. 9905, Springer, Cham, Switzerland, pp. 21–37, Oct. 2016.

[28] Intel, “Intel® Distribution of OpenVinoTM Toolkit,” https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html. Accessed: 2024-03-06.

[29] A. Khan, K. Khattak, Z. H. Khan, T. A. Gulliver, W. Imran, and N. Minallah, “Internet-of-Video Things Based Real-Time Traffic Flow Characterization,” *EAI Endorsed Transactions Scalable Information Systems*, vol. 8, no. 33, art. e9, Oct. 2021.

[30] Intel, “Intel® Neural Compute Stick 2,” https://www.intel.com/content/www/us/en/developer/articles/tool/neural-compute-stick.html. Accessed: 2024-03-06.

[31] J. Lee, B. Varghese, and H. Vandierendonck, “ROMA: Run-Time Object Detection to Maximize Real-Time Accuracy,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, Waikoloa, HI, USA, pp. 6405–6414, Jan. 2023.

[32] University of Idaho, “Traffic Flow Theory, Greenshield’s Model,” https://www.webpages.uidaho.edu/niatt_labmanual/chapters/trafficflowtheory/theoryandconcepts/GreenshieldsModel.htm. Accessed: 2024-03-06.

[33] F. Porikli and O. Tuzel, “Object Tracking in Low-Frame-Rate Video,” *Proceedings of SPIE*, vol. 5685, pp. 72-79, Mar. 2005.

[34] S. Srivastava, A. V. Divekar, C. Anilkumar, I. Naik, V. Kulkarni, and V. Pattabiraman, “Comparative Analysis of Deep Learning Image Detection Algorithms,” *Journal of Big Data*, vol. 8, no. 1, art. 66, May 2021.

[35] Raspberry Pi, “Raspberry Pi 4,” https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. Accessed: 2023-06-22

[36] Raspberry Pi, “RPi Camera Module 2,” https://www.raspberrypi.com/products/camera-module-v2/. Accessed: 2023-06-22.

[37] Intel, “Intel® Neural Compute Stick 2 and Half-Precision Floating Point (FP16),” https://www.intel.com/content/www/us/en/developer/articles/technical/intel-neural-compute-stick-2-intel-ncs-2-and-16-floating-point-fp16.html. Accessed: 2023-06-22.

[38] A. Rpsebrock, “Simple Object Tracking With OpenCV,” https://pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/. Accessed: 2023-06-22.

[39] N. A. Ahmed, “Mean Average Precision (mAP): A Complete Guide,” https://kili-technology.com/data-labeling/machine-learning/mean-average-precision-map-a-complete-guide. Accessed: 2023-06-22.