4-12-2023

Deep Learning Based Vehicle Classification

Ragheb Barkat Muhib
University of Windsor

Deep Learning Based Vehicle Classification

By

Raghib Barkat Muhib

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2023

Deep Learning Based Vehicle Classification

by

Raghib Barkat Muhib

APPROVED BY:

E. Abdel-Raheem
Department of Electrical and Computer Engineering

D. Wu
School of Computer Science

I. Ahmad, Advisor
School of Computer Science

March 24, 2023

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Vehicle classification is an essential part of intelligent transportation systems(ITS). This work proposes a model based on transfer learning, combining data augmentation for the recognition and classification of local vehicle classes in Canada. It takes inspiration from contemporary deep learning (DL) achievements image classification. This makes use of the Dataset named Stanford AI of Vehicles, which has 16185 photos. The images in this section are divided into 196 types of various vehicles. To increase performance further, additional classification blocks are added to the residual network (ResNet-50)-based model which is being used. In this case, vehicle type details are automatically extracted and classified. A number of measures like accuracy, precision, recall, etc. were used during the analysis to evaluate the results. The proposed model exhibited increasing accuracy despite the vehicles' different physical characteristics. In comparison to the current baseline method and the two pre-trained DL systems, AlexNet and VGG-16, our suggested method outperforms them all. The suggested ResNet-50 pre-trained model achieved an accuracy of 90.07% in the classification of native vehicle types, according to outcome comparisons. We have also compared this by running VGG-16 where we are getting an accuracy of 82.5%. Along with this Vehicle classification, we have implemented number plate detection and smart vehicle counter systems which all together makes our transport system better than ever before.

DEDICATION

I would like to dedicate this thesis to my mother for her incredible love and support. Because I believe that she is the real backbone of our family, this is to appreciate her selfless hard work and efforts towards the family.

Furthermore, I dedicate it to my father to raise me like a son and give me wings to fly. To my grandfather, for always trusting me and supporting me in my hard times, without his encouragement, nothing would have been easy. And to my entire family for their unconditional affection towards me.

ACKNOWLEDGEMENTS

I would like to sincerely express my most profound gratitude towards my supervisor Dr.Imran Ahmad, whose input helped me immensely. With his input, I was able to look at my research with a different perspective and a more critical eye.

Secondly, I would like to express my gratitude to my thesis committee members for their beneficial advice and suggestions for my thesis.

I would also like to thank my friends for always encouraging and supporting me.

I humbly extend my thanks to the School of Computer Science and all concerned people who helped me in this regard.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DL	Deep Learning
AI	Artificial Intelligence
NN	Neural Network
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
GPU	Graphics Processing Unit
ANPD	Automatic Number Plate Detection
LPR	License Plate Recognition
ANPR	Automatic Number Plate Recognition
OpenCV	Open Source Computer Vision
VEDA	Vertical Edge Detection Algorithm
ULEA	Unwanted Line Elimination Algorithm
GAP	Global Average Pooling
OCR	Optical Character Recognition
MAP	Mean Average Precision

CHAPTER 1

Introduction

1.1 Introduction

Intelligent Transportation Systems (ITS) refer to the use of technology and information systems to improve the efficiency, safety, and sustainability of transportation systems. ITS encompasses a wide range of technologies, such as advanced traffic management systems, real-time information systems, and autonomous vehicles. The goal of ITS is to make transportation safer, more efficient, and more accessible, while reducing congestion, emissions, and energy consumption. ITS leverages advances in communication, sensor, and computational technologies to provide real-time information and feedback to drivers, operators, and decision-makers, helping to optimize the performance of transportation networks and improve the overall travel experience. The integration of ITS with other smart city technologies is also becoming increasingly important for the development of sustainable and livable cities. In automated transportation systems, a vehicle classification system is used to determine the type of vehicle it observes, based on its size, weight and other features. Vehicle classification is commonly used in intelligent transportation systems.

1.1.1 Deep Learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—allowing it to "learn" from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Many artificial intelligence (AI) apps and services are powered by deep learning, which enhances automation by carrying out mental and physical tasks without the need for human intervention. Deep learning is the technology that enables both established and emerging technologies, like voice-activated TV remote controls, digital assistants, and credit card fraud detection.

How are they different if deep learning is a subset of machine learning? The kind of data it uses and the learning strategies it uses set deep learning apart from traditional machine learning. Structured, labeled data is used by machine learning algorithms to produce predictions, which means that the model's input data is used to identify certain features that are then arranged in tables. This doesn't necessary mean that it doesn't employ unstructured data; it just means that if it does, it generally goes through some pre-processing to put it into a structured format. Some of the data pre-processing that is generally involved with machine learning is eliminated with deep learning. These algorithms can handle text and visual data that is unstructured and automate feature extraction, reducing the need for human specialists. Let's imagine, for instance, that we wanted to categorize a collection of images of various pets by "cat" ,"dog", "hamster" etc. Deep learning algorithms can decide which characteristics like ears are most crucial for differentiating one species from another[41]. This hierarchy of features is created manually by a human specialist in machine learning. The deep learning algorithm then fine-tunes and adapts itself for accuracy through the processes of gradient descent and backpropagation, enabling it to make predictions about a fresh animal shot with greater accuracy. Along with being capable of supervised learning, unsupervised learning, and reinforcement learning, machine learning and deep learning models can also learn in other ways. To categorize or make predictions, supervised learning uses labeled datasets; this involves some sort of human interaction to accurately label input data. Unsupervised learning, in contrast, does not require labeled datasets; instead, it analyzes the data for patterns and groups them according to any identifying traits. A model learns through the process of reinforcement learning to perform an activity in an environment more accurately in order to maximize the reward[68]. Vehicle classification is the process of identifying and categorizing different types of vehicles based on their characteristics, such as make, model, different years, shape, and color, among others. The objective of vehicle classification is to automatically categorize vehicles into predefined classes or categories, such as sedans, SUVs, trucks, buses, and motorcycles, among others. Vehicle classification can be used for various applications, including traffic management, autonomous driving, and intelligent transportation systems. The classification can be performed using various techniques, including traditional computer vision methods and more recent deep learning methods, which have shown superior performance in recent years. Vehicle classification plays a crucial role in understanding the characteristics of different types of vehicles, which can be used to optimize transportation systems and improve safety, efficiency, and sustainability.

1.1.2 Neural Networks

Artificial neural networks, also known as deep learning neural networks, make an effort to imitate the human brain through the use of data inputs, weights, and bias. Together, these components accurately identify, categorize, and characterize items in the data. Deep neural networks are made up of many layers of interconnected nodes, each of which improves upon the prediction or categorization made by the one underneath it. Forward propagation refers to the movement of calculations through the network. A deep neural network's visible layers are its input and output layers. The deep learning model ingests the data for processing in the input layer, and the final prediction or classification is performed in the output layer. Back propagation is a different method that uses techniques like gradient descent to calculate prediction errors before changing the function's weights and biases by iteratively going back through the layers in an effort to train the model. A neural network can make predictions and make necessary corrections for any faults thanks to forward propagation and backpropagation working together. The algorithm continuously improves in accuracy over time[62].

In the simplest terms possible, the aforementioned summarizes the simplest kind of

deep neural network. To solve certain issues or datasets, there are various forms of neural networks, but deep learning techniques are highly complex. For instance, Convolutional neural networks (CNNs), which are mostly employed in computer vision and image classification applications, are able to recognize patterns and features in an image, enabling tasks like object recognition or detection. For the first time in an object recognition test in 2015, CNN outperformed a human. As they make use of sequential or time series data, recurrent neural networks (RNNs) are frequently utilized in applications for speech and natural language recognition[6].
An enormous amount of computational power is needed for deep learning. High performance graphics processing units (GPUs) are the best choice since they have enough of memory and can do lots of computations in several cores. However, managing numerous GPUs on-premises can put a significant strain on internal resources and be prohibitively expensive to grow[58].

1.1.3 Segmentations

A vehicle classification system is one part of a larger system that also includes a recognition subsystem and a tracking subsystem. This study has been segmented in three parts. Those are :

1. Vehicle model detection
2. Vehicle Number plate detection.
3. Vehicle counter

1.1.3.1 Vehicle Model Detection

Vehicle classification with deep learning is a process of identifying and categorizing different types of vehicles using artificial neural networks. The deep learning algorithms use large amounts of data and computational resources to learn and extract features from images or videos of vehicles, and then make predictions about their types. The classifier can be trained on a variety of features, such as make, model,

year, shape, and color, and can be used for various applications, including traffic management, autonomous driving, and intelligent transportation systems. The accuracy of the classification depends on the quality of the training data, the architecture of the neural network, and the optimization techniques used for training. Many applications have been developed for intelligent traffic systems such as parking systems, traffic monitoring, etc. There are three types of patterns that are often used for vehicle type classification. The pattern of vehicle type to determine vehicle type such as micro bus, SUV sedan, hatchback, etc. The vehicle classification pattern based on the logo to find out the vehicle brand such as BMW, Toyota, Volkswagen, etc. Finally, the vehicle classification sets to determine the series of vehicles[27].
In modern cities, cameras for surveillance are nearly universally placed. Real-time observation and event searching are the primary goals of implementing surveillance systems. Police personnel might use the surveillance system to accomplish their search objectives[67], for instance, to look for a specific vehicle. In order to identify a car, officers typically need information on the vehicle's attributes, such as its model and number plate[61]. The police frequently spend a lot of time by themselves watching recorded videos. They must repeat the searching operation numerous times because, in most cases, the searching time exceeds the length of the film. Additionally, after a while of searching, the officers may make certain errors due to fatigue[38].

A variety of sensors are used by traffic management and information systems to estimate traffic characteristics. Counting the number of cars that pass across magnetic loop detectors is a common practice. Systems for video monitoring that rely on vision have several benefits. A far wider range of traffic characteristics, including vehicle classes, lane changes, etc., may be monitored in addition to vehicle counts. In addition, installing cameras is far less disruptive than installing loop detectors. The calculation of the percentages of vehicle classes that use state-aided streets and roads depends on vehicle categorization. Outdated data are used to represent the current condition, and frequently, human operators used to manually count the number of cars on a particular roadway[12].

1.1.3.2 Automatic Number Plate Detection

Automatic Number Plate Detection (ANPD) is a computer vision task that involves identifying and extracting license plate numbers from images or videos of vehicles. The goal of ANPD is to accurately detect and recognize the license plate numbers in real-time, even under challenging conditions such as low lighting, partial occlusion, or irregular font sizes and styles[64].

In our system, the traffic scene is captured by a single camera that is installed on a pole or other tall structure. It may be utilized in any direction of traffic flow and for identifying and categorizing cars in numerous lanes. Only the camera calibration settings and traffic direction are needed to initialize the system[33]. Due to the daily increase in automobiles, license plate recognition (LPR) is more important in today's crowded environment. Vehicle theft, traffic infractions, and encroachment into confined spaces are all rising exponentially, therefore registration code recognition is meant to stop this behavior. Among the key process phases, such as number plate detection, character segmentation, and character recognition, segmentation plays a crucial role since it determines how well each character is recognized. Numerous algorithms have been created for this work in order to minimize issues like unwanted light and tilt that decrease segmentation and, in turn, influence recognition accuracy. In this study, a powerful approach for character localization, segmentation, and identification inside the localized plate is presented. Images captured from still cameras or videos are converted into grayscale versions. In order to reduce the amount of linked parts and compute the connected portion, the Hough transform is used to identify the Hough lines. Grey scale images are segmented by looking for edges for picture smoothing. The registration code's last character is identified. The intention is to demonstrate that by optimizing a number of characteristics that have a greater recognition rate than the conventional approaches, the intended strategy was able to reach high accuracy[22]. Our system is able to detect the blue license plates of Ontario, Canada.

1.1.3.3 Vehicle Counter System

A Vehicle Counter System is a technology used for counting the number of vehicles passing through a specific location, typically a road or a bridge. The system is used to collect traffic data and monitor traffic flow patterns for various purposes such as traffic management, congestion control, and road planning.

A typical Vehicle Counter System consists of two or more sensors installed at the designated location, which detect the presence of a vehicle and record its passing. The data collected by the sensors is processed by an electronic unit that counts the number of vehicles passing in each direction, and the data can be transmitted in real-time to a central location for analysis and storage.

Vehicle Counter Systems can be classified into several types based on the method of vehicle detection, such as inductive loop detectors, microwave radar sensors, video cameras, and infrared sensors. The choice of technology depends on the specific requirements of the application, such as accuracy, cost, and installation requirements.

Overall, Vehicle Counter Systems play an important role in traffic management and road planning by providing accurate and reliable data on traffic flow patterns. This information can be used to optimize traffic signal timings, improve road designs, and make data-driven decisions on transportation planning.

The focus of this study is the introduction of a vision-based system for classifying and counting road vehicles along with the number plate detection. Even in challenging situations associated with occlusions and/or the presence of shadows, the system is able to perform counting with a very excellent degree of accuracy. The system operates on the basis of using already-installed cameras in road networks without the need for extra calibration. In this thesis, we provide a reliable segmentation approach that can identify foreground pixels that belong to moving cars. The method starts by modeling each backdrop pixel with an adjustable Gaussian distribution. In order to accurately locate moving vehicles in both space and time, this model is used in conjunction with a motion detection approach. Occlusions between cars and trucks rise as a result of the trials' nature, which includes peak times and different vehicle kinds. Based on the idea of solidity, a special approach for severe occlusion detection has been developed and put to the test. The technique created in this study can also manage shadows with excellent resolution. The associated algorithm has been put to the test and contrasted with a conventional approach. According to experimental findings based on four sizable datasets, our technology outperforms traditional inductive loop detectors in real time vehicle classification and counting with a high degree of performance (98%) in a variety of environmental conditions [25].

1.1.4 Models Used

1.1.4.1 Part 1: Vehicle Model Detection

We are using ResNet50 for our Vehicle classification model. ResNet-50 is a convolutional neural network that is 50 layers deep. You can load a pre trained version of the network trained on more than a million images from the ImageNet database. The pre-trained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals[47].

1.1.4.2 Part 2 :Vehicle Number Plate Detection

For the second part of our study, we are using ANPR for our number plate detection. Automatic number-plate recognition (ANPR) is a technology that uses optical character recognition on images to read vehicle registration plates to create vehicle location data. It can use existing closed-circuit television, road-rule enforcement cameras, or cameras specifically designed for the task. ANPR is used by police forces around the world for law enforcement purposes, including to check if a vehicle is registered or licensed. It is also used for electronic toll collection on pay-per-use roads and as a method of cataloging the movements of traffic, for example by highways agencies [31].

1.1.4.3 Part 3 : Vehicle Counter System

Finally, for the last part of our study we are using OpenCV for the vehicle counter system. OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products[60].

1.2 Problem Statement and Motivation

The objective of the system is to accurately classify vehicles based on their type and/or characteristics, such as size, shape, and color. This information can then be used for a variety of purposes, such as traffic management, toll collection, law enforcement, and autonomous vehicles. Intelligent Transportation System (ITS) is an advanced application which aims to provide innovative services relating to different modes of transport and traffic management and enable users to be better informed and make safer, more coordinated, and 'smarter' use of transport networks[43].

The categorization of vehicle types is essential for the development of an Intelligent Transport System (ITS). We introduce a model that combines data augmentation with transfer learning for the identification and categorization of local vehicle classes in Canada. By reducing traffic issues, intelligent transportation systems (ITS) seek to improve traffic efficiency.

1.2.1 Challenges and Objectives

The challenges of the vehicle classification system include:

1. Large variability in vehicle appearance: Vehicles can come in a wide range of shapes, sizes, and colors, making it difficult to accurately classify them based

on visual features alone.

2. Occlusions and viewpoint variations: Vehicles may be partially obscured by other objects, such as trees or buildings, which can make it difficult to accurately classify them. Similarly, the viewpoint from which the vehicle is observed can also affect its appearance, making it difficult to accurately classify it.

3. Inconsistent lighting conditions: Vehicles may be observed under varying lighting conditions, such as sunlight, shadows, and nighttime, which can affect their appearance and make it difficult to accurately classify them.

4. Overcoming traditional limitations of hand-crafted features: Traditional methods of vehicle classification often rely on hand-crafted features, which can be time-consuming to develop and are prone to errors.

The vehicle classification system must overcome these challenges to accurately classify vehicles, and the solution must be scalable and efficient, as it will be processing large amounts of data from a variety of sources.

1.2.2 Contribution

We have created a model that is more efficient to classify vehicles than the models that have been used before. Moreover, it can automatically detect number plates and can keep count of the passing vehicles. It is a full system of vehicle classification with number plate detection and vehicle counter. Our Vehicle classification system contributes to the below mentioned issues. Those are summarized as follows:

1. Improved traffic management: By accurately classifying vehicles, a vehicle classification system can provide valuable information to traffic engineers, which can be used to improve traffic management and control.

2. More efficient toll collection: In electronic toll collection systems, accurate vehicle classification is crucial for determining the correct toll fee. A vehicle classification system can improve the efficiency and accuracy of toll collection.

3. Enhanced law enforcement: By accurately classifying vehicles, a vehicle classification system can assist law enforcement agencies in enforcing traffic laws, detecting stolen vehicles, and monitoring compliance with registration and insurance requirements.

4. Improved safety for autonomous vehicles: In autonomous vehicles, accurate vehicle classification is crucial for making informed decisions about how to navigate the road and avoid potential hazards. A vehicle classification system can improve the safety of autonomous vehicles.

5. Improved efficiency: By automating the vehicle classification process and reducing the reliance on manual, time-consuming methods, a vehicle classification system can improve the efficiency of the process and reduce the potential for errors.

6. Advancement in technology: A vehicle classification system that utilizes deep learning techniques represents a significant advance in the field of computer vision and artificial intelligence. It demonstrates the capabilities of these technologies to solve real-world problems, and it paves the way for further development and innovation in these fields.

1.2.3 Thesis Organization

In chapter 1 of our study, We gave a detailed introduction to our deep learning based vehicle classification system. After than in chapter 2, We talked about the research we did for this study. We discussed the relevant studies that have been done previously. We have segmented literature review in three parts just like other chapters in this thesis. After that, in chapter 3, We elaborately explained the machine learning techniques we used for our study which are transfer learning, TensorFlow and the machine learning models. We have mentioned the model architecture as well. Moving on to the next chapter which is chapter 4, We have talked about the Methodology of our thesis. We discussed the methodology used in Vehicle classification, automatic number plate detection and vehicle counter system here. Moving on to chapter 5, we talked about the experiments and results we got. We also discussed the dataset that we used. Finally, in chapter 6, we discussed the result analysis, future work and concluded our study.

CHAPTER 2

Related Works

One of the key components of the smart road management system and traffic management system is the categorization of vehicles. The categorization process is significantly impacted by the employment of suitable algorithms. To handle central portion image feature augmentation with non-fixed size input, we propose a deep neural network in this study called a center strengthened convolutional neural network (CS-CNN)[9]. The centerpiece of this suggested design is center enhancement, which uses ROI pooling to extract more features from the image's centre region. Another is our CS-CNN, which combines a ROI pooling layer with a VGG network architecture to produce detailed feature maps. We will contrast our suggested approach with other common deep learning architectures like VGG-s and VGG-Verydeep-16[5].

2.1 Vehicle Model Detection

Computer vision research is actively being done in the field of tracking moving objects in video feeds. [44] describes a real-time system for monitoring traffic characteristics. For tracking automobiles in clogged traffic situations, it combines an occlusion-based approach with a feature-based one. Instead of tracking complete cars, vehicle sub-features are monitored to address occlusions. However, this method requires a lot of processing resources. [52] describes a moving object detection approach that distinguishes automobiles from the backdrop using an adaptive background subtraction methodology. The background is designed as a slowly changing picture sequence that can adjust to variations in the weather and illumination. In a related study mentioned in [15], one camera is used to track and count pedestrians.Background subtraction is used to segment the pictures from the input image sequence. The tracked pedestrians are then assembled from the related locations that resulted. Region splitting and merging are viewed as graph optimization problems. A technique for identifying lane changes made by cars in a traffic scenario is introduced in[45]. The method is similar to that described in [15], with the addition that vehicle trajectories are calculated to identify lane shifts. Despite the abundance of literature on vehicle tracking and detection, relatively little research has been done on vehicle categorization. This is because categorizing vehicles is a challenging issue by nature. Moreover, the process of classifying vehicles involves more than just detection and monitoring. It is challenging to classify automobiles using straightforward factors since there are so many different forms and sizes of vehicles, even within a single category. When numerous categories are requested, this process becomes considerably more challenging. Occlusions, shadows, camera noise, variations in lighting and weather, etc. are common in real-world traffic scenes. Stereo cameras are also infrequently employed for traffic surveillance.As a result, given a single camera view, recovering vehicle metrics like length, breadth, and height becomes much more challenging. Stereo algorithms are not practical for real-time applications because to their intrinsic complexity and the requirement to address the correspondence problem[59]. Infrustructure of our model has been described later.

In [18], the authors describe a vehicle tracking and classification system that can classify moving objects as either cars or people. The vehicles are not further divided into several classes, though.[49] describes a method for classifying objects that makes use of parameterized 3-D models. The system classifies cars in a traffic flow using a 3-D polyhedral model. The technique makes use of a basic car model with a sedan-like design. The basic premise is that automobiles are more prevalent than trucks or other types of vehicles in ordinary traffic scenarios. The University of Reading has put a lot of effort on three-dimensional vehicle tracking and employing 3-D model matching techniques to categorize the tracked cars.In their 3-D model-based tracking, Baker and Sullivan [53] made use of their understanding of camera calibration and the fact that vehicles travel on a plane. A variety of vehicle types, including sedans, hatchbacks, wagons, etc., were built as three-dimensional wireframe models. Then, projections of these models were contrasted with aspects of the picture. This method was expanded in [15] such that the picture characteristics exert stresses on the model. As a result, performance increased and the number of iterations was decreased. Additionally, they employed principal component analysis to minimize the amount of parameters by parameterized models as deformable templates. In order to achieve real-time performance, Sullivan et al. [8] devised a streamlined version of the model-based tracking technique. In a number of studies [54], CNN was employed as a classifier to categorize the colors of vehicles. The technique put out by Chen et al. [37] to identify the color of the vehicle is feature context. They used their dataset, which included 15,601 photos of vehicles and eight different vehicle color classifications, to classify the photographs. They could classify with an accuracy of 90.68%. The authors of [10], [12] used Chen's dataset [13] and CNN. Their findings revealed accuracy rates of 84.47% and 84.6%, respectively. In their car color categorization experiment, they suggested a novel CNN structure called Colornet, which had the greatest accuracy of 85.74%. The structure performed better than GoogleNet [2] and Alexnet [10].

Deep neural network or deep learning algorithms for vehicle recognition and classification were proposed in another study by Zhou et al. [66]. They employed the YOLO [55] detection paradigm for detection. As methodologies for categorization, Alexnet [10] was employed. There are four different classifications available in categorization modules: passenger vs. other, automobiles vs. vans, sedans vs. taxis, and sedans vs. vans vs. taxis. Both structures were applied, and then they were adjusted to work well with the publicly available dataset that was published in [17]. The experimental findings demonstrated accuracy of above 80%.

2.2 Automatic Number Plate detection

This area has been the subject of much exploration. A lot of information from earlier efforts, the most recent developments, and conference papers may be accessed by referring to many trans acts, books, and conference papers. There are several approaches that have been used to locate the location of a license plate[23]. Three key contributions are made by a quick approach for automobile license plate detection (CLPD) that is suggested [40]. In order to improve the speed of the CLPD approach, we first present a quick vertical edge detection algorithm (VEDA) based on the contrast between the grayscale values. An unwanted line elimination method (ULEA) is suggested to improve the picture after binarizing the input image with adaptive thresholding (AT), and after that, the VEDA is used. The second contribution is the processing of extremely low-resolution web camera photos using our suggested CLPD approach. The appropriate plate features are highlighted using color data once the VEDA has identified the vertical edges. The candidate region will next be extracted using logical and statistical techniques. An LP is eventually discovered. The final contribution is a comparison of the Sobel operator and VEDA in terms of processing speed, algorithm complexity, and accuracy. The results demonstrate precise edge detection performance and five to nine times faster processing than Sobel. Although the combination of edge statistics and mathematical morphology in [48] and[50] produced excellent results, it is time-consuming. As a result, [4] utilizes the blockbase technique[30]. A brand-new technique called N row distance is used in n [39]. This technique counts the edges that are present after scanning an image with N rows of space. The license plate is identified if the number of edges exceeds a threshold; otherwise, the threshold must be lowered and the process will be repeated. For basic photographs, this process is quick and produces decent results. The edge-based algorithms in this study have the drawback of being sensitive to undesirable edges, such as noise edges, and of failing when used on complicated pictures. In [6], edges are first identified, and then the Hough transform is used to determine where the license plate is located[46]. When the license plate border is blurry, warped, or the photos have a lot of vertical and horizontal edges, this approach needs too much processing and has trouble extracting the license plate region. [14] Uses a wavelet-transform-based technique to extract the key information needed for license plate localization. Multiple license plates in a picture can be located using this technique.

In [32], symmetry-based techniques are described. Kim Chien found the license plate by using symmetric areas. This approach takes some time and fails to handle rotated or distorted plates[17].

Neural networks were suggested by Park et al. to find license plates[51]. To determine if each window of an image has a license plate or not, neural networks may be employed as filters to examine tiny windows of the picture. Their inputs are HSI values, and the colors are altered under various lighting conditions, which causes several issues. Zimic et al. [69] have used fuzzy logic to address the issue of finding license plates. In order to determine the horizontal and vertical locations of the license plate, the authors created certain logical rules to characterize the license plate and provided membership functions for the fuzzy sets brilliant, dark, and bright and dark sequence. According to authors, the system may be easily modified to find different patterns by expanding or altering the rules. By collecting a collection of vehicle photographs, Hinde Anoual et al. [3] devised Texture Based VLPL Algorithm (TVLPLA). These images are then sent into the computer code, where they are first converted to grayscale images. To improve the quantity plate and its characters, modifications in contrast, brightness, and gamma are made to the best possible levels. The area where a number plate is most likely to be found is then retrieved and masked for use in further processing. Now, letters and numbers are further searched for in the resultant region of interest by repeatedly adjusting the bounding box's coordinates. The output is kept in a computer program, and after each loop, it is verified to see if it contains all of the number plate's digits. The computer code shows the quantity and ends the program's execution when the results fit the criteria specified, allowing the next image to be analyzed. To choose the characters for license plate areas, Matas et al.[29] recommended using the identification of related components and then a machine learning approach.

2.3 Vehicle Counter System

In many aspects of modern daily life, Deep Learning is a well-liked machine learning method. Numerous individuals have been able to create a variety of Deep Learning-based software or systems to assist human operations and activities thanks to its reliable performance and ready-to-use frameworks and architectures. One use of deep learning is in the monitoring of traffic. Numerous activities, including vehicle identification, traffic infraction monitoring, vehicle speed monitoring, etc., may be accomplished utilizing cameras placed in strategic locations along the highways[24]. In this research, we provide a Deep Learning implementation for a vehicle counting system that does not need tracking the movements of the cars. Pretrained model of YOLOv3 is employed in this research because to its high performance and reasonable computational time in object identification, which helps to improve system performance and minimize time in deploying Deep Learning architecture. In order to categorize and count the cars that cross the street, this project intends to develop a straightforward vehicle counting system. In contrast to earlier studies that exclusively counted cars, the counting is based on four different types of vehicles: cars, motorcycles, buses, and trucks. As a result, our suggested solution has the maximum accuracy of 97.72% when counting the cars crossing the road based on video taken by a camera [4].

CHAPTER 3

Machine Learning Techniques

3.1 Transfer Learning

Transfer learning is a machine learning technique that enables a model trained on one task to be applied to a related but different task, by reusing some of its knowledge. The idea behind transfer learning is that the features learned by a model on one task can be useful for another task, reducing the amount of data and computational resources required to train the model from scratch.

For example, a deep learning model trained on a large dataset of natural images can be used as a starting point to train a model for a different vision task, such as object detection or segmentation. By fine-tuning the pre-trained model on the new task, the model can learn to perform the new task using the features learned from the original task, which provides a good initialization for the new task. This can be particularly useful when the amount of data available for the new task is limited, or when the computational resources required to train a model from scratch are not available.

In summary, transfer learning enables models to leverage knowledge gained from previous tasks to improve performance on new tasks, making machine learning more accessible and efficient. Transfer learning (TL) is a research problem in machine learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. For example, knowledge gained while learning to recognize cars could apply when trying to recognize trucks. From the practical standpoint, reusing or transferring information from previously learned tasks for the learning of new tasks has the potential to significantly improve the sample efficiency of a reinforcement learning agent[57].

3.2 TensorFlow

TensorFlow is a popular open-source software library for machine learning and deep learning. It was developed by the Google Brain team and is used by many companies and researchers around the world. TensorFlow was designed to be flexible, scalable, and efficient, making it a popular choice for building and training machine learning models.

TensorFlow is based on the concept of tensors, which are multidimensional arrays of data, and provides a high-level API for defining, executing, and visualizing computation graphs. The computation graph defines the mathematical operations performed on the tensors, allowing TensorFlow to optimize and parallelize the computations across multiple processors and GPUs. TensorFlow also includes tools for profiling and debugging the computation graph, making it easier to identify and fix performance bottlenecks.TensorFlow provides a wide range of tools for building and training machine learning models, including support for deep neural networks, convolutional neural networks, recurrent neural networks, and more. It also includes a large collection of pre-trained models and tutorials that can be used as a starting point for building custom models.TensorFlow can be used for a variety of tasks, such as image classification, object detection, natural language processing, and generative models, among others. It also provides support for distributed training, making it possible to train models on large datasets using multiple GPUs and computers. A free and open-source software library for artificial intelligence and machine learning is called TensorFlow. Although it may be used for many different tasks, deep neural network training and inference are given special attention.The Google Brain team created TensorFlow for use in internal Google research and production. 2015 saw the first release under the Apache License 2.0.TensorFlow 2.0, the upgraded version of TensorFlow from Google, was launched in September 2019.Python, JavaScript, C++, and Java are just a few of the programming languages that support TensorFlow.This adaptability allows for a wide range of applications across several industries[1].

3.2.1 Characteristics of TensorFlow

The following are some of the key characteristics of TensorFlow:

1. Flexibility: TensorFlow is highly flexible and allows for both low-level and high-level APIs for building and training machine learning models.

2. Portability: TensorFlow can run on a variety of platforms, including desktop computers, GPUs, and mobile devices, making it a versatile tool for machine learning.

3. Auto differentiation: TensorFlow has a powerful auto differentiation engine that can automatically compute gradients of complex models, making it easier to train deep learning models.

4. Distributed computing: TensorFlow allows for distributed computing, allowing for parallel execution of computations across multiple GPUs or machines, making it possible to scale up training to handle large datasets.

5. Model sharing: TensorFlow has a built-in model sharing mechanism, making it easy to share models and collaborate with others on complex machine learning projects.

6. Visualization: TensorFlow has a visualization tool called TensorBoard, which allows users to visualize and debug their models, as well as monitor their training progress.

7. Large community: TensorFlow has a large and active community of users, including researchers and practitioners, who are constantly contributing to its development and improving its features.

In summary, TensorFlow is a powerful and versatile tool for building and training machine learning and deep learning models, providing a high-level API and a large collection of tools and tutorials to make machine learning more accessible and efficient.

Fig. 3.2.1: TensorFlow

3.3 ResNet50

ResNet50 is a popular convolutional neural network architecture for image classification. It was introduced in 2015 by Microsoft Research and has since become one of the most widely used deep learning models for image recognition tasks. ResNet50 is a residual neural network, meaning it uses residual connections, which allow the network to learn the residual mapping between the inputs and outputs. This allows the network to learn much deeper architectures than traditional feedforward neural networks, making it possible to train very deep networks without suffering from the vanishing gradient problem.

The ResNet50 architecture consists of 50 layers, including convolutional, activation, batch normalization, and pooling layers. It uses residual connections, which allow the network to bypass a portion of the layer inputs directly to the outputs, making it easier for the network to learn identity functions. The residual connections also help to mitigate the vanishing gradient problem, allowing the network to learn much deeper architectures.

ResNet50 is pre-trained on a large dataset of natural images, and the pre-trained weights can be used as a starting point for fine-tuning on custom datasets. This makes it possible to train a high-performing image classification model with a relatively small amount of data, reducing the amount of data and computational resources required for training.

In summary, ResNet50 is a highly effective deep learning architecture for image classification, offering a high level of accuracy and performance. Its residual connections allow it to learn much deeper architectures than traditional feedforward networks, making it possible to train very deep networks with a relatively small amount of data. The pre-trained weights also make it possible to quickly train high-performing models on custom datasets, making it a popular choice for many image recognition tasks.

3.3.1 Resnet50 Architecture

A 50-layer convolutional neural network is called ResNet-50 (48 convolutional layers, one MaxPool layer, and one average pool layer). Artificial neural networks (ANNs) that use residual blocks to build networks are known as residual neural networks.Kaiming et al. in study "Deep Residual Learning for Image Recognition"[35] developed the convolutional neural network (CNN) variant known as ResNet. Applications using computer vision frequently employ CNNs [12] . ResNet-34, a version of the original ResNet design, included 34 weighted layers. By utilizing the idea of shortcut connections, it offered a creative solution for expanding the number of convolutional layers in a CNN without encountering the vanishing gradient issue. A shortcut link turns a conventional network into a residual network by "skipping over" some levels. The VGG neural networks (VGG-16 and VGG-19) served as the foundation for the regular network; each convolutional network has a 33 filter. A ResNet, on the other hand, is simpler and contains fewer filters than a VGGNet. In comparison to a VGG-19 Network's 19.6 billion FLOPs, a 34-layer ResNet can perform at 3.6 billion FLOPs, while a smaller 18-layer ResNet can accomplish 1.8 billion FLOPs. The ResNet architecture abides by two fundamental design principles. First, regardless of the size of the output feature map, there are the same number of filters in each layer. Second, to preserve the temporal complexity of each layer even if the size of the feature map is half, it has twice as many filters[28].

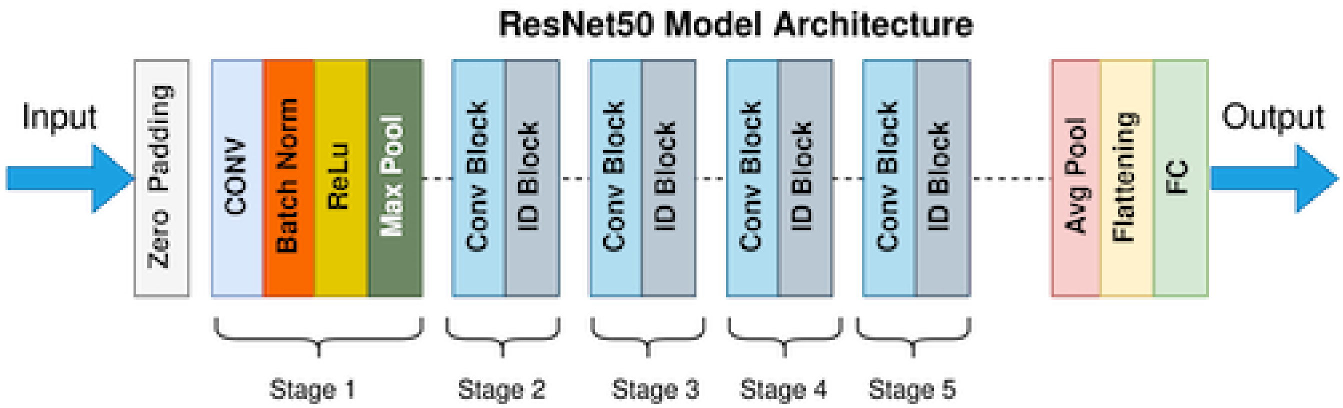

Fig. 3.3.1: ResNet50 Architecture

As listed below, the 50-layer ResNet based design which was changed according to our needs consists of the following components:

1. A 7×7 kernel convolution alongside 64 other kernels with a 2-sized stride.
2. A max pooling layer with a 2-sized stride.
3. 9 more layers—3×3,64 kernel convolution, another with 1×1,64 kernels, and a third with 1×1,256 kernels. These 3 layers are repeated 3 times.
4. 12 more layers with 1×1,128 kernels, 3×3,128 kernels, and 1×1,512 kernels, iterated 4 times.
5. 18 more layers with 1×1,256 cores, and 2 cores 3×3,256 and 1×1,1024, iterated 6 times.
6. 9 more layers with 1×1,512 cores, 3×3,512 cores, and 1×1,2048 cores iterated 3 times.
7. Average pooling, followed by a fully connected layer with 1000 nodes, using the softmax activation function.

3.3.2 Special Characteristics of ResNet50

ResNet50 is fully based on the architecture shown above. Bottleneck building block technology is used in the 50-layer ResNet. The number of parameters and matrix multiplications are decreased while using a bottleneck residual block, which employs 11 convolutions. Every layer's training may now be completed considerably more quickly. Instead of two layers, it makes use of a stack of three layers[19]. ResNet50 is a deep learning architecture for image classification that has several unique characteristics that make it stand out compared to other architectures. These characteristics include:

1. Residual connections: ResNet50 is a residual neural network, meaning it uses residual connections, which allow the network to learn the residual mapping between the inputs and outputs. This allows the network to learn much deeper architectures than traditional feedforward neural networks, making it possible to train very deep networks without suffering from the vanishing gradient problem.

2. Pre-activation design: ResNet50 uses a pre-activation design, where batch normalization and activation functions are applied before the convolutional layer. This allows the network to learn more effectively, as it normalizes the activations before applying the convolutional layer, reducing the covariate shift.

3. Pre-trained weights: ResNet50 is pre-trained on a large dataset of natural images, such as ImageNet, and the pre-trained weights can be used as a starting point for fine-tuning on custom datasets. This makes it possible to train a high-performing image classification model with a relatively small amount of data, reducing the amount of data and computational resources required for training.

4. Performance: ResNet50 has demonstrated excellent performance in image classification tasks, achieving high accuracy and low error rates. It is a popular choice for many image recognition tasks due to its high performance and ease of use.

5. Scalability: ResNet50 is highly scalable, allowing it to be used for a wide range of image classification tasks, from small custom datasets to large-scale image recognition tasks.

3.4 VGG16

The VGG model, commonly known as VGGNet, is referred to as VGG16. It is a 16-layer convolutional neural network (CNN) model. This model was proposed by K. Simonyan and A. Zisserman from Oxford University and presented in the study Very Deep Convolutional Networks for Large-Scale Image Recognition. In ImageNet, a dataset that contains more than 14 million training photos over 1000 item classes, the VGG16 model can reach a test accuracy of 87%. It is a standout model from the 2014 ILSVRC competition[28].

VGG16 enhances AlexNet by substituting sequences of smaller 33 filters for the big filters. For the first convolutional layer in AlexNet, the kernel size is 11, while for the second layer, it is 5. Using NVIDIA Titan Black GPUs, the researchers trained the VGG model over a period of time. The VGG model was initially suggested by Andrew Zisserman and Karen Simonyan in 2013, and a prototype was made for the 2014 ImageNet Challenge. They were a part of Oxford's Visual Geometry Group (VGG)[56].

This approach was different from earlier, successful versions in a number of respects. First of all, AlexNet employed a 1111 receptive field with a 4-pixel stride while it only used a modest 33 receptive field with a 1-pixel stride. The function of a bigger receptive field is achieved by the combination of the 33 filters. When employing numerous smaller layers as opposed to a single big layer, the decision functions are improved and the network can converge more rapidly. This is because there are more non-linear activation layers present.

Secondly, the smaller convolutional filter used by VGG lessens the likelihood that the network would over-fit during training activities. The best size for a filter is 33, as lower sizes can't catch information from the left, right, and up and down. Therefore, VGG is the simplest model that may be used to comprehend the spatial properties of a picture. The network is simple to control thanks to consistent 33 convolutions[65].

3.4.1 VGG16 Architecture

VGG16 is a 16-layer deep neural network, as suggested by its name. With 138 million parameters in all, the VGG16 network is therefore rather large by today's standards. The fundamental draw of the VGGNet16 design is, however, its simplicity.

The most crucial characteristics of convolutional neural networks are included in the VGGNet architecture. VGG networks are made up of tiny convolution filters. VGG16 comprises 13 convolutional layers and three fully linked layers.

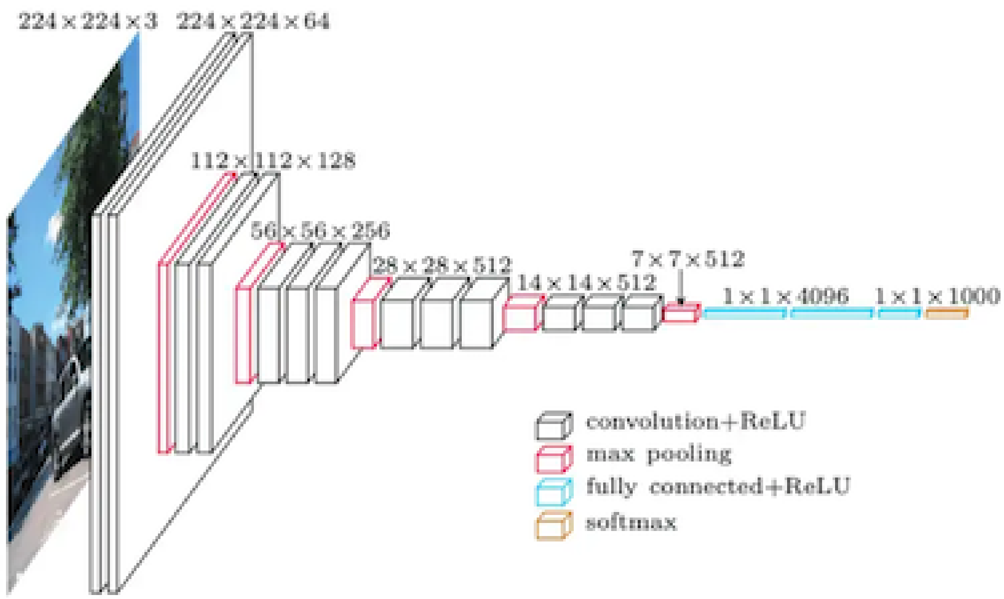

Fig. 3.4.1: VGG16 Architecture

An overview of the VGG architecture is provided below:

1. Input—VGGNet receives a 224×224 image input. In the ImageNet competition, the model's creators kept the image input size constant by cropping a 224×224 section from the center of each image.

2. Convolutional layers—the convolutional filters of VGG use the smallest possible receptive field of 3×3. VGG also uses a 1×1 convolution filter as the input's linear transformation.

3. ReLu activation—next is the Rectified Linear Unit Activation Function (ReLU) component, AlexNet's major innovation for reducing training time. ReLU is a linear function that provides a matching output for positive inputs and outputs zero for negative inputs. VGG has a set convolution stride of 1 pixel to preserve the spatial resolution after convolution (the stride value reflects how many pixels the filter "moves" to cover the entire space of the image).

4. Hidden layers—all the VGG network's hidden layers use ReLU instead of Local Response Normalization like AlexNet. The latter increases training time and memory consumption with little improvement to overall accuracy.

5. Pooling layers–A pooling layer follows several convolutional layers—this helps reduce the dimensionality and the number of parameters of the feature maps created by each convolution step. Pooling is crucial given the rapid growth of the number of available filters from 64 to 128, 256, and eventually 512 in the final layers.

6. Fully connected layers—VGGNet includes three fully connected layers. The first two layers each have 4096 channels, and the third layer has 1000 channels, one for every class[42].

3.5 VGG16 vs Resnet

To increase accuracy, VGG proposed the idea of adding more layers. The model may not converge, though, if the number of layers is increased beyond 20. The vanishing gradient issue is the primary cause; after an excessive number of folds, the learning rate is so low that the model's weights remain static. Gradient explosion is a different problem. Gradient clipping is a method for updating the weights that requires "clipping" the error derivative to a predetermined threshold during backward propagation. Rescaling the weights together with the error derivative lowers the likelihood of an overflow or underflow, which can cause a gradient explosion. The Residual Network (ResNet) architecture uses the concept of skip connections, allowing inputs to "skip" some convolutional layers. The result is a significant reduction in training time and improved accuracy. After the model learns a given feature, it won't attempt to learn it again—instead, it will focus on learning the new features. It's a clever approach that can significantly improve model training[21].

3.5.1 Difference Between VGG16 and ResNet50

VGG16 and ResNet50 are both popular deep learning architectures for image classification. However, there are several key differences between them:

1. Architecture: VGG16 has a simple and shallow architecture, consisting of 16 layers with only 3x3 convolutional filters, while ResNet50 has a deeper architecture, consisting of 50 layers with a mix of 1x1, 3x3, and 5x5 convolutional filters.

2. Depth: ResNet50 is designed to handle the vanishing gradient problem that can occur in very deep networks, whereas VGG16 does not have this ability. This allows ResNet50 to learn much deeper architectures than VGG16.

3. Residual connections: ResNet50 uses residual connections to learn the residual mapping between the inputs and outputs, allowing the network to train much deeper architectures, while VGG16 does not use residual connections.

4. Performance: In general, ResNet50 has higher accuracy compared to VGG16 on large image datasets like ImageNet, due to its deeper architecture and ability to handle the vanishing gradient problem.

5. Speed: VGG16 is faster to train than ResNet50 due to its shallower architecture, but ResNet50 can still be trained in a reasonable amount of time on modern hardware.

In brief, VGG16 is a simple and fast architecture for image classification, while ResNet50 is a deeper and more accurate architecture for the same task. The choice between the two depends on the specific requirements of the task and the trade-off between accuracy and speed.

CHAPTER 4

Methodology

4.1 Vehicle Model Detection

The ResNet-50 architecture is utilized as the basis model in this study for classifying picture data, and we updated it to increase accuracy. Fig. 4.1.1 illustrates the system's flow. Additionally, we compared the best architecture we obtained in earlier research utilizing the VGG Network model, ResNet models, and CNN architecture. In our suggested approach, we base our model on residual networks (ResNets). One deep convolutional network that employs a connection shortcut to address vanishing gradient problems is ResNets. ResNet uses shortcut connection as an identity function to deepen the architecture. This block is referred to as a residual block.

Fig. 4.1.1: Vehicle Model Detection Methodology

The ResNet-50 architecture's residual block is made up of three convolutional layers with 11, 33, and 11 filters. On each convolution layer that is created with the identity shortcut, batch normalization is done, followed by relu activation. If the feature map's dimensions are the same as the connection shortcut and the sequence of three convolutional layers, the summing between them may be performed. The initial convolution layers impart a stride of 2 during the downsampling process, changing the size of the feature map and making it impossible to make a shortcut link. Giving a convolution layer with a kernel size of 11 and stride of 2 on the shortcut connection to match the dimensions can solve this problem. The Global Average Pooling layer completes the ResNet-50 design (GAP). GAP uses softmax activation for classification and averages each feature map before sending it to the fully connected layer procedure [7].

4.2 Number Plate detection

Every country now has a serious issue with traffic regulation and car ownership identification. It might be challenging to recognize car owners who drive excessively fast and against the regulations of the road. As a result, it is sometimes tough to apprehend and penalize those individuals since traffic officials in local areas may not be able to obtain the car's license plate from a moving vehicle due to its speed without their advanced devices. As one of the answers to this problem, it is necessary to design an automatic number plate detection (ANPD) system [36]. ANPD typically involves the following steps:

1. Image Pre-processing: This involves converting the input image or video frame into a format that can be processed by the algorithm. This often includes operations such as converting the image to grayscale, filtering, and resizing.

2. License Plate Detection: This step involves locating the license plate region in the image or video frame. This can be done using techniques such as edge detection, region proposal algorithms, or machine learning-based approaches.

3. Character Segmentation: Once the license plate region has been located, the next step is to segment the individual characters within the license plate. This can be done using techniques such as morphological operations, connected component analysis, or deep learning-based segmentation methods.

4. Character Recognition: After the individual characters have been segmented, the next step is to recognize the characters. This can be done using techniques

such as optical character recognition (OCR), machine learning-based recognition methods, or deep learning-based recognition models.

Overall, ANPD is a challenging computer vision task that requires a combination of image processing, machine learning, and computer vision techniques to accurately and efficiently detect and recognize license plate numbers. there are several steps in number plate detection which we followed for our system. Those are :

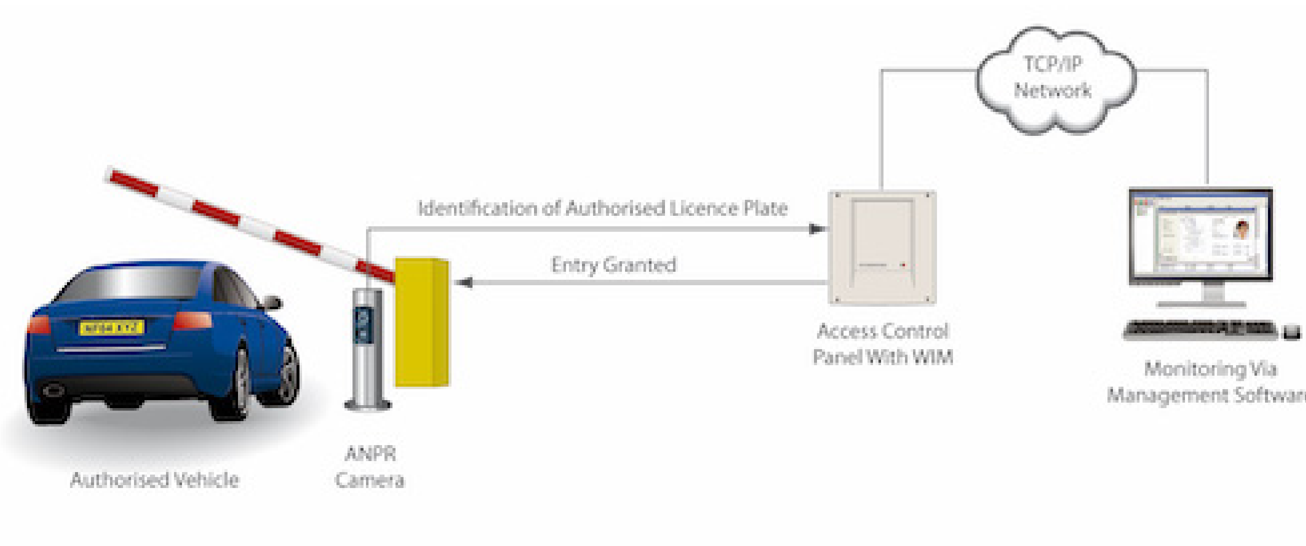

Fig. 4.2.1: Number Plate detection

1. Step 1 : Read Image, Grayscale and Blur. Gray scaling allows us to convert from a specific color code to a different one.

Fig. 4.2.2: ANPR step 1

2. Step 2 : Apply filter and find edges. Remove noises and detect the edges

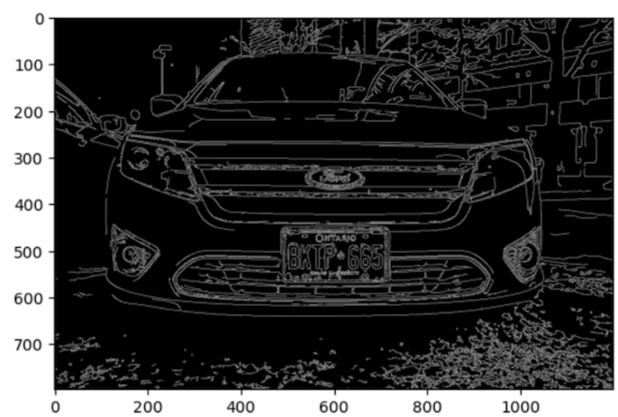

Fig. 4.2.3: ANPR step 2

3. Step 3 : Finding Contour and Applying Mask. Firstly, the system finds shapes. Then it returns a tree of contours to traverse what kind of shapes we're looking for. If the keypoint is 4 , that is most likely our number plate. To isolate the particular shape from the image the system applies mask. Finally, it draws a contour within the image. In this case the specific location.

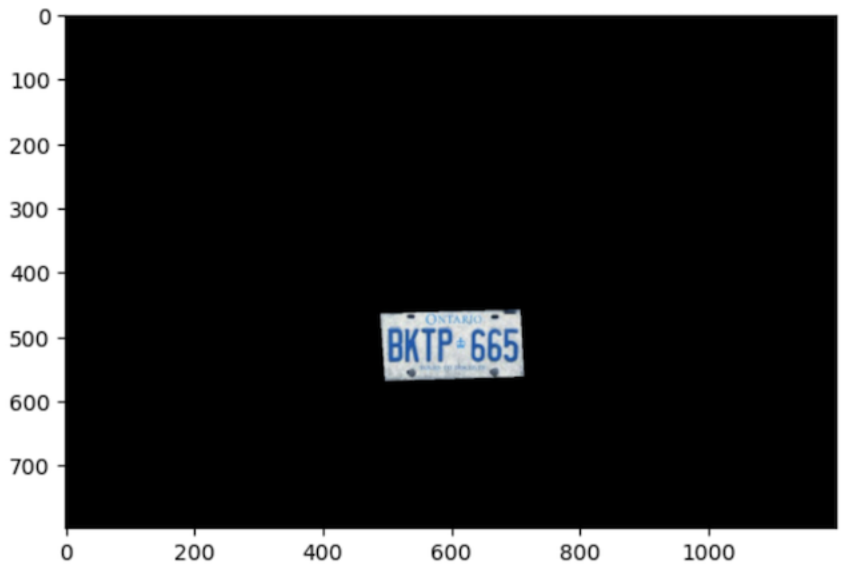

Fig. 4.2.4: ANPR step 3

4. Step 4 : Use Easy OCR to Read Text and Render Result. Optical Character Recognition (OCR) is the process that converts an image of text into a machine-readable text format. For example, if you scan a form or a receipt, your computer saves the scan as an image file. You cannot use a text editor to edit, search, or count the words in the image file. Use Easy OCR and get the number of the Vehicle and Get the final result of ANPR[11].

Fig. 4.2.5: ANPR step 4 and 5

4.3 Vehicle Counter System

A vehicle counting system typically consists of the following steps:

1. Image acquisition: The first step involves capturing an image or a video of the scene where vehicles are to be counted. The image should be captured from a high and unobstructed viewpoint.

2. Image pre-processing: In this step, the captured image or video is processed to remove noise, adjust the contrast, and perform any necessary correction.

3. Vehicle detection: In this step, the image is processed to detect and locate the vehicles in the image. This can be done using a variety of methods, such as background subtraction, blob analysis, or deep learning-based object detection.

4. Vehicle tracking: In this step, the detected vehicles are tracked across multiple frames to ensure that the same vehicle is not counted multiple times.

5. Vehicle counting: Finally, the number of vehicles passing through a specified region is counted and recorded.

6. Data analysis: The recorded data can be analyzed to obtain various statistics, such as the number of vehicles passing through a particular region at a specific time or the average speed of the vehicles[23].

Note that the exact steps and methods used in a vehicle counting system may vary depending on the specific application and requirements. We modified our system based on our need. This part of our study is the most simple which is done in few steps. Given are the following steps :

1. Step 1 : Read Input image frame with OpenCV.

2. Step 2 : Using TensorFlow Object detection API

3. Step 3 : Detect Vehicle Image

4. Step 4 : Vehicle Counting

CHAPTER 5

Experiments and Results

In this study, our main focus in on Vehicle classification with Resnet50. We have also used the same dataset to compare the result with another state of the art model VGG16 and other resNet models.

5.1 Dataset

The Stanford AI Cars dataset is a valuable resource for researchers working on computer vision and machine learning problems, such as object detection, semantic segmentation, and fine-grained recognition. The dataset is publicly available and has been widely used in a variety of research projects and publications.

Fig. 5.1.1: Stanford AI Cars

There are 16,185 photos of 196 different kinds of automobiles in the Cars collection. The data has been divided into 8,144 training photos and 8,041 testing images, roughly splitting each class 50-50. Typically, classes are organized by Make, Model, and Year, for example, 2012 Tesla Model S or 2012 BMW M3 coupe. We can say that the Stanford AI Cars dataset is an important benchmark for computer vision and machine learning research, providing a rich source of data for training and evaluating algorithms in the field.

5.2 Experimentation

Here, We tried two methods:

Approach 1 (two-stage approach): To obtain the car bounding box from a picture, first utilize a car detection model. After that, crop the car out of the picture before subjecting it to an image classification system. The overall throughput of this process was high since this method performed feature extraction twice.

Approach 2 (single-stage approach): Classifying cars by first detecting them using single-stage detectors like RetinaNet. The accuracy was comparable to the two-stage approach with this method, but my throughput time was reduced to a very outstanding 0.30 seconds.

5.3 Creativity

1. Loss Function: I am using a weighted form of object detection loss to focus the training more on classification rather than object detection. Object Detection is important for extracting the correct feature from the image but here I felt even with low IoU, we should get the correct classification.

The loss function that I am using is:

$$Loss = \ classification\_weight * FocalLoss + regression\_weight * L1Loss$$

Where Focal Loss is given as:

$FL = -\ y(1-p)^{y} \log p - \ (1-y)p^{y} \log(1-p)$

2. Fully Convolutional Network: The network is fully convolutional. This enables resizing of images by scale. This is very important as I think scale will play a key

distinguishing factor in differentiating between some categories. 3. Accuracy Metric: I am using Mean Average Precision as my accuracy metric. Its a function of both precision and recall.
It is given as:

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}$$

4. Cyclic Training The training of retinanet was done in 4 iterations:

Training Iteration 1:

$Hyperparameters : \ Epochs = 20, Ł1\_loss\_weight = 1.0, \ Focal\_loss\_weight = 1.0, \ Learning\_rate = 0.01 \ , MAP = 0.198$

Training Iteration 2:

$Hyperparameters : \ Epochs = 20, Ł1\_loss\_weight = 0.8, \ Focal\_loss\_weight = 1.2, \ Learning\_rate = 0.01 \ , MAP = 0.252$

Training Iteration 3:

$Hyperparameters : \ Epochs = 10, Ł1\_loss\_weight = 0.5, \ Focal\_loss\_weight = 2.0, \ Learning\_rate = 0.001 \ , MAP = 0.285$

Training Iteration 4:

$Hyperparameters : \ Epochs = 10, Ł1\_loss\_weight = 0.2, \ Focal\_loss\_weight = 5.0, \ Learning\_rate = 0.001 \ , MAP = 0.319$

5.4 Different experiments and results analysis

5.4.1 Initial Result

At the beginning of our experiments we split the dataset in half for training and testing purposes. After that we trained the dataset against ResNet50 and VGG16 for getting a good comparison. Our initial result is given below: At that point of our study, we found that ResNet50 got a validation accuracy of 84.5% and Testing accuracy of 83.05%. On the other hand, VGG16 gave us a validation accuracy of 81%

Architecture	Validation Accuracy	Testing Accuracy
VGG16	81%	79%
ResNet50	84.5%	83.05%

Table 5.4.1: Initial Result

and testing accuracy of 79%. It is highly visible that ResNet50 is giving us better accuracy.

5.4.2 Final Result

As part of our experiments we changed the ratio of training data and testing data to see some different results. In the next part of our experiment we kept 80% of the dataset for training and 20% of the dataset for testing purposes. After training our model with this ratio of dataset we got better results for both ResNet50 and VGG16. The results are given below :

Finally after we are done with our experiments with ResNet50 and VGG16 we got

Architecture	Validation Accuracy	Testing Accuracy
VGG16	82.5%	79.6%
ResNet50	90.07%	85.5%

Table 5.4.2: Final Result

the result shown in the table above. We got 90.07% validation accuracy and 85.5% testing accuracy for ResNet50. Although we got better accuracy with ResNet50, there are still some misclassifications in our model. For example, Audi A5 Coupe was misclassified as Chevrolet TrailBlazer SS, Audi S6 Sedan was misclassified as Mitsubishi Lancer Sedan, Dodge Challenger SRT8 was misclassified as Chevrolet Camaro Convertible, Ferrari 458 Italia Coupe was misclassified as Geo Metro Convertible. These are the few misclassifications for which we couldn't achieve higher than the result we got. However, we got 82.5% validation and 79.6% testing accuracy for VGG16. Even though VGG16 might be faster sometimes, ResNet50 gives better results each and every time.

5.4.3 Comparison With Different Models

To compare ResNet50 with other resNet models , we used the same dataset with the ratio of 4:1 which means 80% for Training data and 20% for testing data. We are comparing with ResNext101, ResneXt50, ResNet152V2, MobileNetV2 the comparison is given below :

Architecture	Validation Accuracy	Testing Accuracy
ResNeXt101	78%	79.02%
ResXeXt50	77.7%	76.03%
ResNet152V2	78.02%	78%
MobileNetV2	71.3%	80.05%

Table 5.4.3: Comparison Results

To compare our achieved result with other models, we kept the experimental environment the same and got the result given in the table 5.4.3.

5.4.3.1 ResNext101

ResNeXt101 is a variant of the ResNet (Residual Network) architecture designed to improve the accuracy of deep neural networks. It is characterized by the use of grouped convolutions, where multiple parallel convolutional layers are used to extract features from the input data, instead of using a single, large convolutional layer. The number “101” in the name refers to the number of layers in the network.ResNeXt101 was introduced in the study “Aggregated Residual Transformations for Deep Neural Networks”[63] and has been shown to achieve state-of-the-art results on a variety of computer vision tasks such as image classification and object detection. Overall, ResNeXt101 can be seen as a deeper and wider version of ResNet, incorporating more layers and more filters in each layer, which allows it to learn more complex representations from the input data. After our experiment with this model, we got validation accuracy of 78% and testing accuracy of 79.02% .

5.4.3.2 ResNeXt50

ResNeXt50 is a variant of the ResNeXt architecture, which is an improvement over the ResNet architecture. The number "50" in the name refers to the number of layers in the network.Like other Resnet models, ResNeXt50 uses grouped convolutions to extract features from the input data, where multiple parallel convolutional layers are used instead of a single, large convolutional layer. This allows the network to learn more complex representations of the input data and improve its accuracy. ResNeXt50 has been shown to achieve state-of-the-art results on a variety of computer vision tasks such as image classification and object detection. Additionally, the smaller size of ResNeXt50 compared to other ResNeXt models makes it computationally more efficient and easier to train, while still providing good performance. Overall, ResNeXt50 is a deep neural network architecture designed to improve the accuracy and efficiency of deep learning models for computer vision tasks. However, the results we got from our experiment after using this model didn't meet our expectations. ResNeXt50 gave a validation accuracy of 77.7% and testing accuracy of 76.03%.

5.4.3.3 ResNet152V2

ResNet152V2 is a variant of the ResNet (Residual Network) architecture that was introduced in the paper "Identity Mappings in Deep Residual Networks". The number "152" in the name refers to the number of layers in the network, and the "V2" refers to the fact that it is the second version of the ResNet152 architecture. ResNet152V2, like other ResNet models, uses residual connections to allow the network to learn complex representations of the input data by adding a shortcut connection that bypasses one or more layers. This helps to address the vanishing gradient problem, which is a common issue in deep neural networks. Compared to its predecessor ResNet152, ResNet152V2 has a more simplified structure with fewer parameters and flops, making it more computationally efficient. However, despite its smaller size, ResNet152V2 still provides good performance on a variety of computer vision tasks, such as image classification and object detection. Overall, ResNet152V2 is a deep neural network architecture designed to improve the accuracy and efficiency of deep learning models for computer vision tasks. For ResNet152V2 we found validation and testing accuracy of 78.02% and 78%.

5.4.3.4 MobileNetV2

MobileNetV2 is a variant of the MobileNet architecture, which is a light-weight deep neural network designed for mobile and embedded devices with limited computational resources. The "V2" in the name refers to the fact that it is the second version of the MobileNet architecture. MobileNetV2 is designed to strike a balance between accuracy and efficiency, with a focus on reducing the number of computations required to perform inference. This is achieved through the use of depthwise separable convolutions, which break down a standard convolutional layer into two simpler operations: a depthwise convolution, which applies a filter to each input channel separately, and a pointwise convolution, which combines the results of the depthwise convolution into a compact representation. This structure allows MobileNetV2 to have significantly fewer parameters and computations compared to other deep neural networks, while still providing good performance on a variety of computer vision tasks, such as image classification and object detection. Overall, MobileNetV2 is a light-weight deep neural network architecture designed for efficient deployment on mobile and embedded devices, while still providing good accuracy for computer vision tasks. However, our experimental result with this particular model is average compared to other models we mentioned earlier. They gave us a testing accuracy of 80.05% but the validation accuracy is as low as 71.3%.

While experimenting with these models, it was seen that the training and testing takes up a lot more time than ResNet50 and VGG16 . However the results are not as good as ResNet50. After comparing all these results with our ResNet50 model, we can see that ResNet50 is more accurate than any other state of the art models seen over the years. We conclude that ResNet50 gives better results than other models that have been used before.

5.5 Additional Features

5.5.1 Web Server

A web server is a software system that delivers web pages over the Internet. When a user requests a web page using a web browser, the browser sends a request to the server, which then sends back the requested web page.

The web server is responsible for managing multiple requests from different users and delivering the appropriate web pages in response. It uses the Hypertext Transfer Protocol (HTTP) to communicate with web browsers, and it can also handle other protocols such as Secure HTTP (HTTPS) and File Transfer Protocol (FTP).

In addition to serving web pages, a web server can also host other types of content, such as images, videos, and audio files, and can run dynamic web applications that generate content on the fly based on user input.

Common examples of web servers include Apache, Flask, Nginx, and Microsoft IIS. Web servers are essential components of the World Wide Web, and they are used by organizations of all sizes, from small businesses to large corporations. We are using flask for our system. Flask is a web framework for Python that provides a simple and lightweight way to build web applications. It was created by Armin Ronacher in 2010 and has since become one of the most popular web frameworks for Python. Flask is known for its simplicity and ease of use, making it a good choice for small to medium-sized projects and for developers who are just getting started with web development.

Flask provides a minimal set of tools for building web applications and leaves the rest to be handled by third-party libraries. This allows developers to choose the tools that best fit their needs and provides a high level of flexibility. Flask uses the Jinja2 template engine for rendering HTML, and it supports the use of RESTful APIs.

Flask applications can be easily deployed to a variety of hosting platforms, including cloud-based services like Heroku, AWS, and Google Cloud. Overall, Flask is a flexible, easy-to-learn, and popular choice for building web applications with Python.

A flask server is a collection of one or more computers that are packed together and solely dedicated to running software applications over the internet. It is server software that can handle HTTP requests on the public internet, private LANs, and private WANs[34].

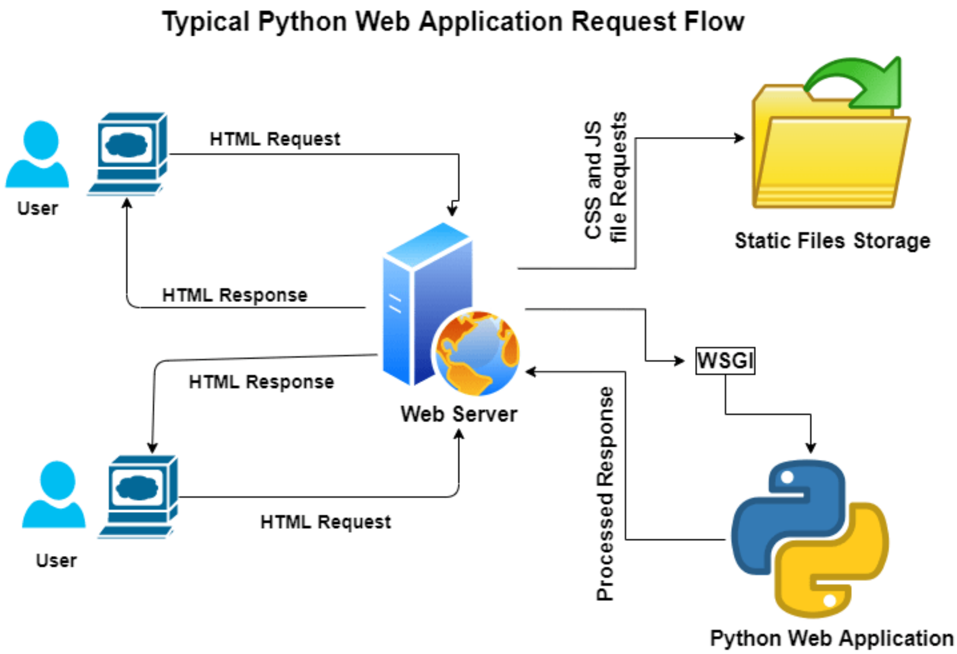

Fig. 5.5.1: Web server

5.5.2 User Interface

Vue.js is a progressive JavaScript framework for building user interfaces and single-page applications. It was created by Evan You in 2014 and has since grown into one of the most popular JavaScript frameworks in the world.

Vue.js is known for its simplicity, performance, and reactivity, making it a popular choice for building dynamic and interactive web applications. The framework provides a set of tools for building reusable components, handling data binding and rendering, and managing the state of your application.One of the key features of Vue.js is its reactivity system, which allows the framework to automatically update the UI when the underlying data changes. This makes it easy to build dynamic and interactive applications without having to write complex logic for handling data updates. Vue.js also provides a set of tools for building custom directives, filters, and plugins, which allow developers to extend the functionality of the framework. In addition, Vue.js has a rich ecosystem of third-party plugins and tools, such as the Vue CLI, which provides a set of tools for quickly scaffolding and deploying Vue.js applications. Overall, Vue.js is a popular and widely-used JavaScript framework for building dynamic and interactive web applications. Its simplicity, performance, and reactivity make it a popular choice for developers of all skill levels[13]. We used Vue.Js for our responsive UI.

Fig. 5.5.2: Vue Js

The majority of the typical functionalities required in frontend development are covered by the ecosystem and framework known as Vue. However, the web is incredibly diverse, and the things we create there may differ greatly in scale and shape. In light of this, Vue is made to be adaptable and gradual in its adoption. Vue may be utilized in a variety of ways, depending on your use case:

1. Improving static HTML without a build step
2. Embedding as Web Components on any page
3. Application That Is Just One Page (SPA)
4. Fullstack / Server-Side Rendering (SSR)
5. Jamstack / Static Site Generation (SSG)

6. Targeting PC, mobile, WebGL, and even the terminal

Fig. 5.5.3: UI

Our final UI looks like the image above.

CHAPTER 6

Discussion, Future Work and Conclusion

6.1 Discussion

Vehicle classification has become an increasingly important area of study in recent years, particularly with the advent of deep learning techniques. This is because the ability to accurately classify vehicles has numerous applications in a variety of industries, from traffic management and toll collection to law enforcement and autonomous vehicles.

One of the primary benefits of vehicle classification is improved traffic management. By accurately classifying vehicles, traffic engineers can better understand the flow of traffic and make informed decisions about traffic control and road design. For example, if a particular road is frequently congested with heavy vehicles, traffic engineers can implement measures to manage the flow of these vehicles, such as creating dedicated lanes or increasing the frequency of traffic lights. Similarly, if a particular road is frequently used by slow-moving vehicles, such as bicycles, traffic engineers can design roadways that are safer and more accommodating for these types of vehicles.

Another important application of vehicle classification is in electronic toll collection systems. In these systems, accurate vehicle classification is crucial for determining the correct toll fee. For example, a heavy commercial truck might be charged a different toll fee than a small passenger car. By accurately classifying vehicles, electronic toll collection systems can ensure that each vehicle is charged the correct fee, which helps to improve the efficiency and accuracy of the toll collection process.

Vehicle classification is also important for law enforcement purposes. By accurately classifying vehicles, law enforcement agencies can monitor and enforce traffic laws more effectively. For example, they can enforce speed limits, monitor compliance with lane usage laws, and detect vehicles that are not properly registered or insured. Additionally, vehicle classification can be used to identify stolen vehicles, which is important for both law enforcement and insurance companies.

If we talk more, another important application of vehicle classification is in autonomous vehicles. In order for autonomous vehicles to operate safely and effectively, they must be able to accurately classify other vehicles on the road. For example, they must be able to distinguish between a passenger car and a truck, and between a pedestrian and a bicycle. By accurately classifying vehicles, autonomous vehicles can make informed decisions about how to navigate the road and avoid potential hazards.

Deep learning techniques have revolutionized the field of vehicle classification, as they allow for more accurate and efficient classification of vehicles. With deep learning, computers can automatically learn the features that distinguish different types of vehicles, such as the shape, size, and color of a vehicle. This allows for more accurate classification than traditional methods, which often rely on hand-crafted features and are prone to errors.

6.2 Future Work

We have demonstrated a model-based system for vehicle categorization, number plate recognition, and vehicle counting that operates reliably in the majority of situations. The system requires little scene-specific knowledge and is sufficiently broad to be able to identify, track, and categorize cars. As long as a vehicle is visible, the system offers position and velocity data in addition to vehicle categorization.

We plan to categorize cars using a nonrigid model-based method to enable categorization into more categories. We'll utilize parameterized 3-D models of examples from each category. A 2-D projection of the model will be created at this angle based on the camera calibration. To identify the vehicle's class, this projection will be compared to the ones in the image.

6.3 Conclusion

In conclusion, vehicle classification is an important area of study with numerous applications in a variety of industries. With the advent of deep learning techniques, it has become possible to achieve more accurate and efficient vehicle classification, which has numerous benefits for traffic management, toll collection, law enforcement, and autonomous vehicles.

REFERENCES

[1] Abadi, M. (2016). Tensorflow: learning functions at scale. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, pages 1–1.

[2] Anand, R., Shanthi, T., Nithish, M., and Lakshman, S. (2020). Face recognition and classification using googlenet architecture. In *Soft Computing for Problem Solving: SocProS 2018, Volume 1*, pages 261–269. Springer.

[3] Anoual, H., El Fkihi, S., Jilbab, A., and Aboutajdine, D. (2011). Vehicle license plate detection in images. In *2011 International Conference on Multimedia Computing and Systems*, pages 1–5. IEEE.

[4] Awang, S. and Azmi, N. M. A. N. (2018). Vehicle counting system based on vehicle type classification using deep learning method. In *IT Convergence and Security 2017: Volume 1*, pages 52–59. Springer.

[5] Chen, M. and Chang, T. (2011). A parking guidance and information system based on wireless sensor network. In *2011 IEEE International Conference on Information and Automation*, pages 601–605. IEEE.

[6] Chen, W., Sun, Q., Wang, J., Dong, J.-J., and Xu, C. (2018). A novel model based on adaboost and deep cnn for vehicle classification. *Ieee Access*, 6:60445–60455.

[7] Collado, J. M., Hilario, C., De la Escalera, A., and Armingol, J. M. (2004). Model based vehicle detection for intelligent vehicles. In *IEEE Intelligent Vehicles Symposium, 2004*, pages 572–577. IEEE.

[8] Cott, B., Durham, R., Lee, P., and Sullivan, G. (1989). Process model-based engineering. *Computers & chemical engineering*, 13(9):973–984.

[9] Dickson, C., Wallace, A. M., Kitchin, M., and Connor, B. (2013). Improving infrared vehicle detection with polarisation.

[10] Espinosa, J. E., Velastin, S. A., and Branch, J. W. (2017). Vehicle detection using alex net and faster r-cnn deep learning models: a comparative study. In *Advances in Visual Informatics: 5th International Visual Informatics Conference, IVIC 2017, Bangi, Malaysia, November 28–30, 2017, Proceedings 5*, pages 3–15. Springer.

[11] Gnanaprakash, V., Kanthimathi, N., and Saranya, N. (2021). Automatic number plate recognition using deep learning. In *IOP Conference Series: Materials Science and Engineering*, volume 1084, page 012027. IOP Publishing.

[12] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[13] Hong, P. (2018). *Practical web design: Learn the fundamentals of web design with HTML5, CSS3, bootstrap, jQuery, and vue. js.* Packt Publishing Ltd.

[14] Hsieh, C.-T., Juan, Y.-S., and Hung, K.-M. (2005). Multiple license plate detection for complex background. In *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)*, volume 2, pages 389–392. IEEE.

[15] Hu, C., Bai, X., Qi, L., Chen, P., Xue, G., and Mei, L. (2015). Vehicle color recognition with spatial pyramid deep learning. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2925–2934.

[16] Jin, J. S., Xu, C., Xu, M., Peng, Y., Jin, J. S., Luo, S., Xu, M., Au, S., Zhang, Z., and Cui, Y. (2013). Vehicle type classification using data mining techniques. In *The era of interactive media*, pages 325–335. Springer.

[17] Kim, D.-S. and Chien, S.-I. (2001). Automatic car license plate extraction using modified generalized symmetry transform and image warping. In *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No. 01TH8570)*, volume 3, pages 2022–2027. IEEE.

[18] Kim, H. (2022). Multiple vehicle tracking and classification system with a convolutional neural network. *Journal of Ambient Intelligence and Humanized Computing*, 13(3):1603–1614.

[19] Koonce, B. and Koonce, B. (2021). Resnet 50. *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, pages 63–72.

[20] Krause, J., Stark, M., Deng, J., and Fei-Fei, L. (2013). 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia.

[21] Kumar, A., Razi, R., Singh, A., and Das, H. (2020). Res-vgg: A novel model for plant disease detection by fusing vgg16 and resnet models. In *Machine Learning, Image Processing, Network Security and Data Sciences: Second International Conference, MIND 2020, Silchar, India, July 30-31, 2020, Proceedings, Part II 2*, pages 383–400. Springer.

[22] Kumthekar, A., Owhal, M. S., Supekar, M. S., and Tupe, M. B. (2018). Recognition of vehicle number plate using raspberry pi. *International Research Journal of Engineering and Technology*, 5(4):391–394.

[23] Lee, D., Yoon, S., Lee, J., and Park, D. S. (2016). Real-time license plate detection based on faster r-cnn. *KIPS Transactions on Software and Data Engineering*, 5(11):511–520.

[24] Lei, M., Lefloch, D., Gouton, P., and Madani, K. (2008). A video-based realtime vehicle counting system using adaptive background method. In *2008 IEEE*

International Conference on Signal Image Technology and Internet Based Systems, pages 523–528. IEEE.

[25] Leone, A. and Distante, C. (2007). Shadow detection for moving objects based on texture analysis. *Pattern Recognition*, 40(4):1222–1233.

[26] Li, D., Liang, B., and Zhang, W. (2014). Real-time moving vehicle detection, tracking, and counting system implemented with opencv. In *2014 4th IEEE international conference on information science and technology*, pages 631–634. IEEE.

[27] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110.

[28] Mascarenhas, S. and Agarwal, M. (2021). A comparison between vgg16, vgg19 and resnet50 architecture frameworks for image classification. In *2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON)*, volume 1, pages 96–99. IEEE.

[29] Matas, J. and Zimmermann, K. (2005). Unconstrained licence plate and text localization and recognition. In *Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005.*, pages 225–230. IEEE.

[30] Matrone, F., Grilli, E., Martini, M., Paolanti, M., Pierdicca, R., and Remondino, F. (2020). Comparing machine and deep learning methods for large 3d heritage semantic segmentation. *ISPRS International Journal of Geo-Information*, 9(9):535.

[31] Mercy, S. S. G., Muthulakshmi, I., and Scholar, P. (2018). Automatic number plate recognition using connected component analysis algorithm. *International Journal For Technological Research In Engineering*, 5(7).

[32] Mittal, A. and Sharma, M. (2008). Number plate recognition using analytical fourier mellin transform. In *Novel Algorithms and Techniques in Telecommunications, Automation and Industrial Electronics*, pages 37–42. Springer.

[33] Modwel, G., Mehra, A., Rakesh, N., and Mishra, K. K. (2021). A robust real time object detection and recognition algorithm for multiple objects. *Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science)*, 14(1):331–338.

[34] Mufid, M. R., Basofi, A., Al Rasyid, M. U. H., Rochimansyah, I. F., et al. (2019). Design an mvc model using python for flask framework development. In *2019 International Electronics Symposium (IES)*, pages 214–219. IEEE.

[35] Mukti, I. Z. and Biswas, D. (2019). Transfer learning based plant diseases detection using resnet50. In *2019 4th International conference on electrical information and communication technology (EICT)*, pages 1–6. IEEE.

[36] Parvin, S., Rozario, L. J., and Islam, M. E. (2021). Vehicle number plate detection and recognition techniques: a review. *Advances in Science, Technology and Engineering Systems Journal*, 6:423–438.

[37] Peng, Y., Dharssi, S., Chen, Q., Keenan, T. D., Agrón, E., Wong, W. T., Chew, E. Y., and Lu, Z. (2019). Deepseenet: a deep learning model for automated classification of patient-based age-related macular degeneration severity from color fundus photographs. *Ophthalmology*, 126(4):565–575.

[38] Petrovic, V. S. and Cootes, T. F. (2004). Analysis of features for rigid structure vehicle type recognition. In *BMVC*, volume 2, pages 587–596. Kingston University, London.

[39] Prabhakar, P. and Anupama, P. (2014). A novel design for vehicle license plate detection and recognition. In *Second International Conference on Current Trends In Engineering and Technology-ICCTET 2014*, pages 7–12. IEEE.

[40] Prabhakar, P., Anupama, P., and Resmi, S. (2014). Automatic vehicle number plate detection and recognition. In *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, pages 185–190. IEEE.

[41] Psyllos, A., Anagnostopoulos, C.-N., and Kayafas, E. (2011). Vehicle model recognition from frontal view image measurements. *Computer Standards & Interfaces*, 33(2):142–151.

[42] Qassim, H., Verma, A., and Feinzimer, D. (2018). Compressed residual-vgg16 cnn model for big data places image recognition. In *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, pages 169–175. IEEE.

[43] Qureshi, K. N. and Abdullah, A. H. (2013). A survey on intelligent transportation systems. *Middle-East Journal of Scientific Research*, 15(5):629–642.

[44] Rachmadi, R. F. and Purnama, I. (2015). Vehicle color recognition using convolutional neural network. *arXiv preprint arXiv:1510.07391*.

[45] Ramyar, S., Homaifar, A., Karimoddini, A., and Tunstel, E. (2016). Identification of anomalies in lane change behavior using one-class svm. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 004405–004410. IEEE.

[46] Rasheed, S., Naeem, A., and Ishaq, O. (2012). Automated number plate recognition using hough lines and template matching. In *Proceedings of the world congress on engineering and computer science*, volume 1, pages 24–26.

[47] Reddy, A. S. B. and Juliet, D. S. (2019). Transfer learning with resnet-50 for malaria cell-image classification. In *2019 International Conference on Communication and Signal Processing (ICCSP)*, pages 0945–0949. IEEE.

[48] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

[49] Sharma, S. and Guleria, K. (2022). Deep learning models for image classification: comparison and applications. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 1733–1738. IEEE.

[50] Silva, S. M. and Jung, C. R. (2018). License plate detection and recognition in unconstrained scenarios. In *Proceedings of the European conference on computer vision (ECCV)*, pages 580–596.

[51] Song, H., Kim, M., Park, D., Shin, Y., and Lee, J.-G. (2022). Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*.

[52] Su, B., Shao, J., Zhou, J., Zhang, X., and Mei, L. (2015). Vehicle color recognition in the surveillance with deep convolutional neural networks. In *2015 Joint International Mechanical, Electronic and Information Technology Conference (JIMET-15)*, pages 790–793. Atlantis Press.

[53] Sullivan, G. D., Baker, K. D., Worrall, A. D., Attwood, C., and Remagnino, P. (1997). Model-based vehicle detection and classification using orthographic approximations. *Image and vision computing*, 15(8):649–654.

[54] Sultana, F., Sufian, A., and Dutta, P. (2018). Advancements in image classification using convolutional neural network. In *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pages 122–129. IEEE.

[55] Sumit, S. S., Watada, J., Roy, A., and Rambli, D. (2020). In object detection deep learning methods, yolo shows supremum to mask r-cnn. In *Journal of Physics: Conference Series*, volume 1529, page 042086. IOP Publishing.

[56] Takisawa, N., Yazaki, S., and Ishihata, H. (2020). Distributed deep learning of resnet50 and vgg16 with pipeline parallelism. In *2020 Eighth International Symposium on Computing and Networking Workshops (CANDARW)*, pages 130–136. IEEE.

[57] Torrey, L. and Shavlik, J. (2010). Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global.

[58] Tsai, C.-C., Tseng, C.-K., Tang, H.-C., and Guo, J.-I. (2018). Vehicle detection and classification based on deep neural network for intelligent transportation applications. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1605–1608. IEEE.

[59] Tsai, L.-W., Hsieh, J.-W., and Fan, K.-C. (2007). Vehicle detection using normalized color and edge map. *IEEE transactions on Image Processing*, 16(3):850–864.

[60] Uke, N. and Thool, R. (2013). Moving vehicle detection for measuring traffic count using opencv. *Journal of Automation and Control Engineering*, 1(4).

[61] Wen, X.-Z., Fang, W., and Zheng, Y.-H. (2011). An algorithm based on haar-like features and improved adaboost classifier for vehicle recognition. *ACTA ELECTONICA SINICA*, 39(5):1121.

[62] Xiaozhu, X. and Cheng, H. (2017). Object detection of armored vehicles based on deep learning in battlefield environment. In *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, pages 1568–1570. IEEE.

[63] Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2017). Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500.

[64] Ying, K., Ameri, A., Trivedi, A., Ravindra, D., Patel, D., and Mozumdar, M. (2015). Decision tree-based machine learning algorithm for in-node vehicle classification. In *2015 IEEE Green Energy and Systems Conference (IGESC)*, pages 71–76. IEEE.

[65] Zakaria, N., Mohamed, F., Abdelghani, R., and Sundaraj, K. (2021). Vgg16, resnet-50, and googlenet deep learning architecture for breathing sound classification: A comparative study. In *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)*, pages 1–6. IEEE.

[66] Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th*

international conference on computational linguistics: technical papers, pages 2335–2344.

[67] Zhang, X., Zhao, C., Sha, Y., Dang, Q., and Zhang, Y. (2013). Vehicle brand recognition based on hog feature and support vector machine. *Journal of Southeast University*, 43:411–413.

[68] Zhang, Z., Tan, T., Huang, K., and Wang, Y. (2011). Three-dimensional deformable-model-based localization and recognition of road vehicles. *IEEE transactions on image processing*, 21(1):1–13.

[69] Zimic, N., Ficzko, J., Mraz, M., and Virant, J. (1997). The fuzzy logic approach to the car number plate locating problem. In *Proceedings Intelligent Information Systems. IIS'97*, pages 227–230. IEEE.

VITA AUCTORIS

NAME: Raghib Muhib

PLACE OF BIRTH: Bangladesh

YEAR OF BIRTH: 1997

EDUCATION:

University of Windsor, M.Sc in Computer Science, Windsor, Ontario, 2023