8-26-2022 11:00 AM

Towards the development of a cost-effective Image-Sensing-Smart-Parking Systems (ISenSmaP)

Aakriti Sharma, *The University of Western Ontario*

Supervisor: Haque, Anwar, *The University of Western Ontario*
Co-Supervisor: Mohsenzadeh, Yalda, *The University of Western Ontario*
A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

Abstract

Finding parking in a busy city has been a major daily problem in today's busy life. Researchers have proposed various parking spot detection systems to overcome the problem of spending a long time searching for a parking spot. These works include a wide variety of sensors to detect the presence of a vehicle in a parking spot. These approaches are expensive to implement and ineffective in extreme weather conditions in an outdoor parking environment. As a result, a cost-effective, dependable, and time-saving parking solution is much more desirable. In this thesis, we proposed and developed an image processing-based real-time parking-spot detection system using deep-learning algorithms. In this regard, we annotated the images using the Visual Geometry Group (VGG) annotator and preprocessed the dataset using the image contrast enhancement technique that attempts to solve the illumination changes in pictures captured in an open space, followed by training the model using the Mask-R-CNN (Region-Based Convolutional Neural Network) and Faster-RCNN algorithms. ROIs (Regions of interest) are used later to determine the vacancy status of each parking spot. Our experimental results demonstrate the effectiveness of our developed parking systems as we achieved a mean Average Precision (mAP) of 0.999 for the PKLot dataset and a mAP of 0.9758 for custom datasets. Furthermore, as part of the smart parking application, we developed an Android App that can be used by the end users. Our proposed intelligent parking system is scalable, cost-effective, and to the best of our knowledge, it offers higher parking spot detection accuracy than any other solutions in this domain.

Keywords

Autonomous systems, smart parking, machine learning, deep learning, artificial intelligence, Convolutional Neural Network (CNN), Faster- Region-Based Neural Network, Mask- Region Based Neural Network

Summary for Lay Audience

The recent advancement and growth in the automotive industry have significantly increased the number of new vehicles on the road every year. However, this elevation is creating traffic congestion, which increases pressure on existing parking lots capacity in urban areas. Searching for empty parking spaces is time-consuming and has become a major problem in a busy city. Many research works have been developing an intelligent parking spot detection system to address this issue. Most of these research works considered sensor-based solutions. These systems provide accurate results but are expensive to implement and need ongoing maintenance. Besides, the sensors are not very effective in extreme weather conditions in an outdoor parking environment. On the other hand, image-based machine learning and deep learning models have shown the potential to be cost-effective techniques.

This thesis introduces an image-based smart parking solution with deep learning techniques for a real-time parking detection system. This system can detect available parking spots more efficiently and accurately than existing solutions. Our proposed solution and developed prototype can be implemented in real-life parking spot detection, offering cost savings and convenience for the parking lot operators and end users.

Acknowledgments

First and foremost, I would like to thank Dr. Anwar Haque, my supervisor, for providing me with this wonderful opportunity to conduct research in developing smart parking systems using AI techniques. Dr. Haque has been a constant source of encouragement, supervision, and inspiration during my master's degree at Western University. I would also like to thank my co-supervisor Dr. Yalda Mohsenzadeh for her valuable technical input to the thesis.

I would also want to thank my parents and friends for their unwavering support during the current pandemic, which allowed me to continue working on my thesis. I would also like to express my gratitude to all my instructors at Western for their guidance during my studies and teaching assistantship. Each course I registered in and participated in during my graduate studies benefited my research.

Finally, I would like to express my gratitude to Western University's Computer Science department for offering me the opportunity to further improve my intellect and skill sets in preparation for my future career.

Table of Contents

List of Figures

List of Tables

Chapter 1

1 Introduction

1.1 Overview

With recent advancements in Artificial Intelligence (AI), fueled by innovations in advanced wireless network technologies, the growth in autonomous systems and solutions is visible and the impact is felt across all industry segments. With the advancement of such smart technologies, people seek secure and convenient solutions in every walk of their lives. Autonomous systems can complete a job, achieve a goal, or interact with their surroundings with little or no human intervention. It is also critical for these systems to forecast, plan, and be aware of their surroundings[1]. Artificial Intelligence and its underlying machine learning capabilities play a significant role in developing these autonomous systems which are of high interest to every business and household [2]. AI is a subfield of Computer Science that aims to imitate human cognition abilities into machines. AI contains six branches: Machine Learning, Neural Networks, Robotics, Expert systems, Natural Language Processing and Fuzzy Logic[3]. At the same time, Machine Learning and Deep Learning are the techniques used by AI to generate intelligent systems[3]. In contrast, Machine Learning (ML) is a core element of AI. It tries to offer computer knowledge through data, observations, and interaction with the environment, allowing computers to extrapolate to ever-changing conditions accurately [3]. Moreover, Deep Learning (DL) is a subset of ML; it employs a layered framework of algorithms known as an artificial neural network, which learns from data to optimize their layered connections to produce the desired output [3]. So, intelligent systems learn from the training data that they experience and act according to the circumstances (the data) they learned. AI has been used extensively to help develop intrusion detection, facial recognition, object detection, pattern recognition, self-driving cars, security or chatbots to make routine work more effortless. In this thesis, we explored and developed an autonomous parking spot management system using AI techniques in an unstructured data environment.

According to statistics, in the UK, drivers, on average, spend 44 hours a year searching for parking spots (an estimated cost impact of £23.3 billion a year)[6]. It is estimated that about one-third of the vehicles travelling a busy city at any given time are searching for parking spots. Smart parking is one of the intelligent service domains increasing in popularity because of its attractive service offerings such as convenience, safety, time savings, and economic benefit, especially in busy city life. The recent development in 5G technology will further support and trigger the growth of smart parking solutions that needs low latency and high bandwidth for their real-time processing. Existing smart parking systems use various sensors that detect a car's presence in parking lots. Wireless sensor nodes are deployed in the parking spaces, and the status of parking spaces is monitored and updated on the central server through sensor nodes and WI-FI/LTE networks. The above sensor-based smart parking technologies are expensive to maintain, complex to manage, and do not scale well for large parking spaces.

Sensor-based parking detection systems require physical sensors deployed at every node to be able to detect the occupancy of a particular parking spot. The sensory systems have a high accuracy rate but suffer from high installation expenses and ongoing maintenance costs. These systems can be used for smaller parking lots with limited parking spaces. Employing the system over a parking lot with over 300-400 spaces each would increase the total budget cost, as well as will require ongoing maintenance for the sensors. These systems can deteriorate over time and may fail during severe weather conditions. Therefore, a system that can reduce these extra operational expenses and easily gather parking occupancy information and details/features of the occupant(car, truck, person) could be an attractive alternative. In this regard, image-based parking spot detection systems can be used to address the problem. These systems require minimal mechanical equipment like CCTV cameras and can be equipped with AI engine to serve as a smart parking detection application.

Table 1 Counter-Based vs Sensor-Based vs Image-Based Parking Detection Systems

Counter-Based	**Sensor based**	**Image Based**
Ticketed parking systems.	Sensors deployed at every parking spots.	Camera installed at aerial angle view of parking lot.
Cars are counted at Entry and Exit gates.	Sensors tracks and send signals further about spot occupancy.	Image frames of live video are sent after every 3-4 seconds for object detection. Machine Learning and Deep Learning algorithm applied for object detection
Manual labelling of parking lots.	Sensors are connected with raspberry Pi and further information is sent to the server to create the whole detection system.	Frames Received are classified into occupied and vacant parking spaces and further sent to server to create an end-to-end user detection system.
No information about exact available parking	Different types of sensors can be used like: Infrared, Ultrasonic, RFID.	Only CCTV required. Uses different algorithms to create more efficient systems like :CNN, Mask-RCNN, Faster-RCNN
No theft management,	Cannot be saved from theft.	CCTV can also be used for theft management.
Non automated, longer park detection time.	Fully Automated, accurate and precise results.	Fully automated system. Gives results based on visuals. So higher accuracy and precision than both systems compared
No detection results.	Does not detect false positive parking, or illegal parking	Can be modified to detect illegal parking, False Positive results.

Mechanical costs: Gated entrance, Ticket Machine etc.	Sensors are expensive, even if cost effective are being selected cannot be deployed for large parking lots. High cost for maintenance.	Cost effective, Robust, CCTV cameras are easily available and installed. Mechanical cost: Gated entrance, CCTV camera. Very less maintenance cost if compared.
Not automated for detections	Can fail or give wrong results during severe weather conditions.	Can still give decent results if trained over bad weather scenarios, until snow has covered all the visual contents or parking lot(parking lines).

Our proposed idea of detecting parking spot occupancy status from image-based intelligent sensing is novel and addresses the challenges above while offering convenience, safety, and economic benefits.

In this research, we developed a machine learning-based real-time parking-spot recognition system. In this thesis, we investigated Deep Learning algorithms used within the field of computer vision. The research begins with data acquisition for a custom dataset. In this regard, we worked on two parking lot datasets: PKLot(Publicly available parking dataset) and the custom dataset. The datasets were divided into three different parts labelled as training, testing and validation data. First, the images were cleaned, and then annotated to create XML(Extensible Markup Language) and JSON(JavaScript Object Notation) files which are the input to our designed recognition system. The data is being preprocessed and augmented to train the data further to detect the availability. The model is also observed after applying different image augmentations and contrast enhancement techniques to construct a more varied dataset.

Furthermore, two different architectures of CNN were used to segment the parking space occupancy within the parking lot. Real-time image inferencing may be derived from this system: training the network with numerous pictures of the same class can detect its common qualities and classify whether new images belong to a particular class. This is done implicitly by extracting or defining certain features of a class that are common to all members of that class. In the succeeding stage, the model was tested and evaluated on its classification performance. The testing images were obtained from a video source; frames were sampled from the video at equal intervals. These images were then used to check the availability of the parking spots in a parking lot.

Researchers have created different CNN designs, such as Faster-RCNN and Mask-RCNN, to solve the issue of parking spots. The authors [7] applied Mask-RCNN on the PKLot dataset to construct a detection model that recognizes a vehicle and finds the corresponding Intersection over the Union of the slot, giving the result as a vacant or occupied slot. Sairam et al. [8] worked on the Mask-RCNN architecture to construct a system capable of identifying parking lots for the user and providing appropriate parking spaces for two-wheeled vehicles. In contrast, Agrawal et al. [9] proposed the multi-angle parking detection technique. The authors created a model that uses Mask-RCNN to predict the number of automobiles and parking lot occupancy based on several camera viewpoints. In comparison to these models, our research also focuses on different CNN architectures: Faster-RCNN and Mask-RCNN. Our system was implemented on two different datasets to develop a parking detection system. Further exploring Mask-RCNN for detecting slot occupancy and to reduce the time to train and test the model, we used ResNet50 and ResNet101 to perform detection and segmentation tasks on our datasets to achieve a reliable, intelligent parking system.

1.2 Objectives and Contributions

The main goal of this project is to develop a cost-effective, real-time, AI-assisted parking spot detection system that provides high accuracy while requiring lower setup and maintenance costs compared to the existing solutions. The application will detect vacant parking spaces in a parking lot and guide the driver (using a mobile Application) to available parking spots. Our research has the following objectives:

Obj #1: To develop real-time, efficient and intelligent (machine learning-based) parking spot detection algorithms that can detect the occupancy status of parking spots in a parking lot.

Obj #2: To develop a mobile Application (Android) that can create 3-D parking map visuals from parking spot occupancy. This planar view would show information about the parking lot with exact available parking spots in a parking lot.

Here we focused on image processing and deep learning-based object identification. Deep Learning plays an important role in object detection, due to its multi-layer structures which are used for extracting features and detailed information from an image we worked on parking lot assessment using Convolutional Neural Networks(CNNs) to identify available parking spaces. We also implemented several image contrast enhancement strategies, to analyze their outcomes, and then selected the most effective strategy, which can increase the model's precision. The main contributions of this thesis are as follows:

- Our proposed novel image-sensing-smart-parking solution is a real-time image-based parking spot detection system. It detects available parking spaces within a parking lot and then navigates the driver (using a mobile App) to the selected parking space.
- The parking detection system uses image frames gathered from the live video feed of the parking lots. Classification is done over the gathered image frames using Faster-RCNN and Mask-RCNN algorithms.
- We focused on a comparative analysis of different backbones for Mask-RCNN and how they affected training duration and model precision.

- Also, four distinct contrast enhancement approaches were used. Four different datasets were developed using these algorithms. Predictions were producing additional comparisons that may aid in developing the most robust and effective parking detecting system.
- Mask-RCNN model obtained a mAP of 0.9990 on the PKLot dataset.
- Parking detection system using Pre-processed images generates a mAP of 1.0 on PKLot dataset
- We also developed an algorithm for generating planar visuals from parking data. Our developed tool takes the feature set of the classification phase (i.e., parking spot availability data) and generates a visualization for the parking lot displaying the available parking spots.
  - As for the potential impacts of this research, the initiative will provide significant advantages to drivers, parking lot operators, and government/road operators. Potential benefits of our developed smart parking systems are:
  - From the end-user standpoint, it provides significant conveniences - saving time, enhancing productivity, and providing economic gain.
  - An efficient, better controlled and monitored parking system increases the possibility for additional revenue from the parking operator's standpoint. System’s available parking places are promptly disseminated in real-time to automobiles seeking parking. In comparison to other sensor-based systems, this platform is more cost-effective than other solutions. Additionally, the suggested technology has minimal ongoing maintenance expenses and is highly scalable.
  - From the standpoint of the government/municipality/road operator, the approach is environmentally friendly since cars spend less time on the road waiting for parking, decreasing road congestion.
  - Users of the application can also easily access and view the nearby available parking lots and be navigated to the reserved spot using phone navigation or infotainment setup of cars.

1.3 Thesis Outline

The rest of the thesis is organized as follows: Chapter 2 presents the necessary technical background that readers would need to understand the content discussed in this thesis, including the ML and DL algorithms and associated image datasets. Chapter 3 provides a comprehensive literature review on smart parking systems. Our proposed methodology and framework are discussed in Chapter 4. Chapter 5 shows the results observed from the proposed methods. Finally, Chapter 6 concludes our research by summarizing the research model along with future enhancements, followed by references used for this research.

Chapter 2

2 Background

This chapter will briefly review the background topics relevant to this thesis. We will cover four main sections. Section 2.1 will review the field of AI and its branches. Section 2.2 will review the field of machine learning and its key concepts. Section 2.3 will cover the concept of deep learning and architecture. Sections 2.4, 2.5 and 2.6 would discuss about different deep learning algorithms. Section 2.6 describes the fundamentals of Image Processing.

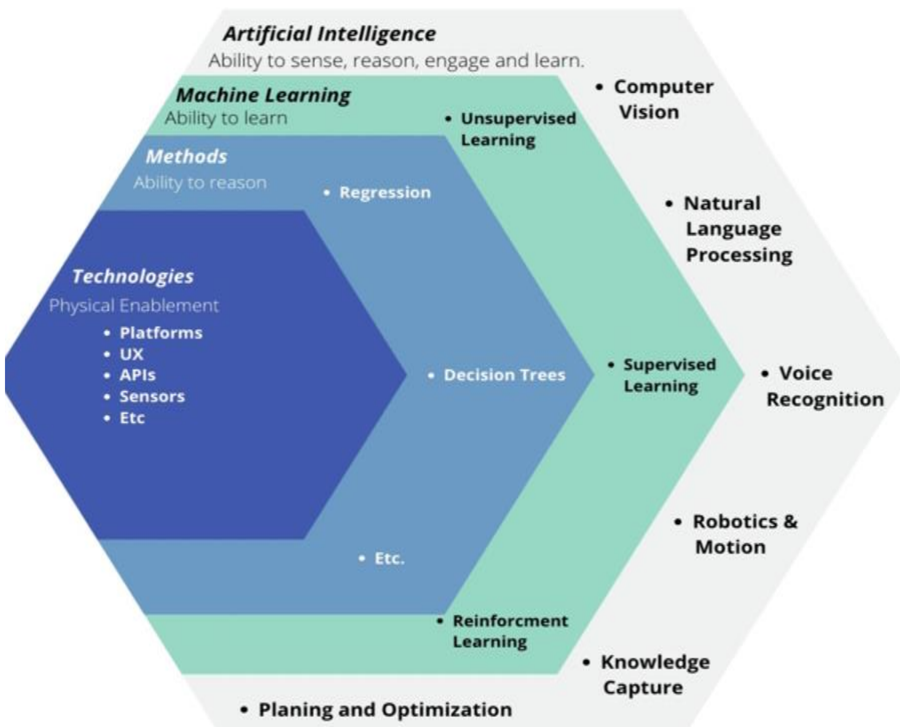


Figure 1 Artificial Intelligence Architecture [66]

2.1 Artificial Intelligence

Artificial Intelligence (AI) is the science and engineering behind creating intelligent machines, particularly innovative computer programs. It is akin to the same goal of utilizing computers to study human intellect, but AI does not have to limit itself to physiologically observable ways[10]. Therefore, AI is the ability of a machine or computer equipment to mimic human cognitive processes, learn from experiences, adapt to new knowledge, and do human-like tasks. It performs tasks intelligently, significantly increasing accuracy, flexibility, and productivity for the entire system. Linguistics, bias, vision, planning, robotic process automation, natural language processing, decision science, and other approaches are included in artificial intelligence as shown in Figure 1.

Machine learning is a branch under the umbrella of AI, and it uses statistics and algorithms to find patterns within datasets. ML is a way to teach the model by training it over datasets to learn the patterns and predict the correct patterns for future data. ML teaches the system to behave and make decisions like humans. ML algorithms use human intervention depending on the circumstances. Supervised Learning, Unsupervised Learning, and reinforcement learning are the three primary machine learning models [3].

A dataset is a collection of data that can be used for teaching and inferencing the machine models to learn and predict the item. A dataset can be structured or Unstructured. Structured data contains meaningful data inside a spreadsheet. Whereas, Unstructured data is usually acoustic, text, video, and image datasets. While working on the Machine Learning model, the dataset is divided into two or three parts: Training, Testing and Validation sets.

Table 2 Comparison between structured and unstructured data training

Structured Data	**Unstructured Data**
Quantitative Data	Qualitative Data
Form of numbers and Values	Form of text, audio, video files.
Data is used in machine learning and to run machine learning algorithms	Data is used in natural language processing and text mining and Computer Vision.

2.2 Machine Learning categories

Machine Learning algorithms are further divided into three categories of Learning: Supervised, Unsupervised and Reinforcement Learning. In this section, we learn about machine learning categories and algorithms. Figure 2 explains the taxonomy of Machine Learning and various techniques and algorithms used within different learning categories.

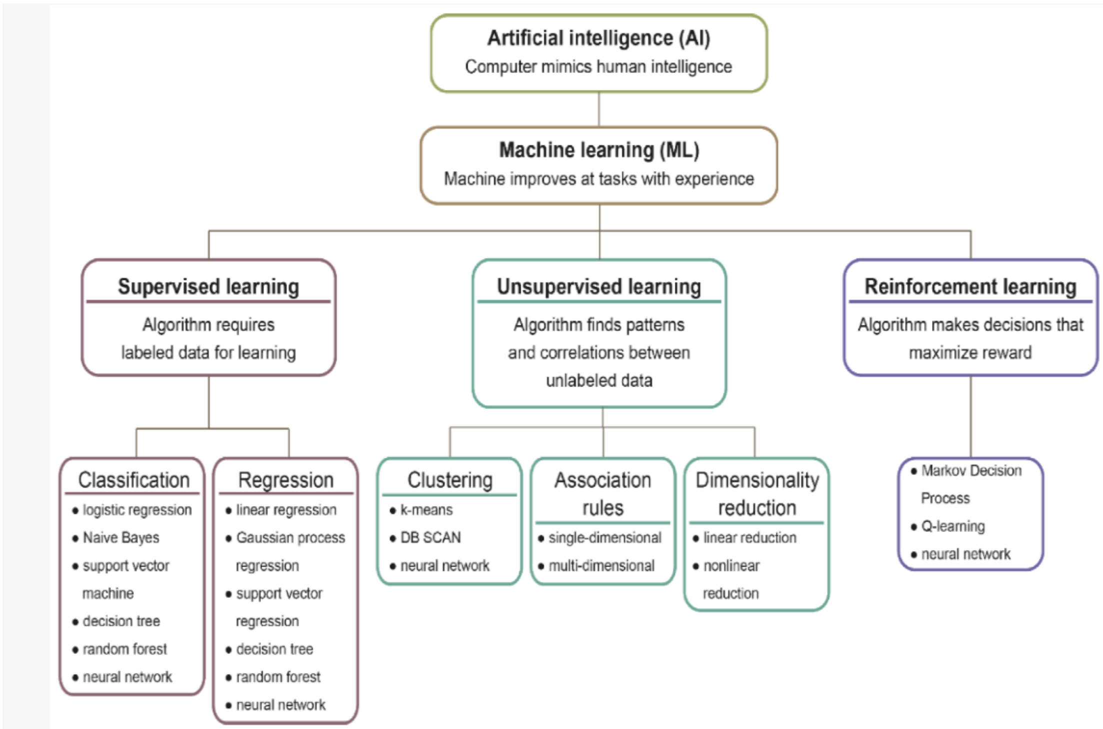


Figure 2 Artificial intelligence and its branches [4]

2.2.1 Supervised Learning

Supervised learning refers to ML algorithms which are trained using well-labelled training data, and the output is predicted based on that data. Labelled data indicates that some input data has already been marked with the appropriate output. The model learns through labelled data on each data type, as shown in Figure 3. After training with all labelled data, the model is tested on a dataset unknown to the system and predicts accordingly [11]. Supervised learning is conducive to solving real-life problems as it has been taught exactly what to look for while predicting.

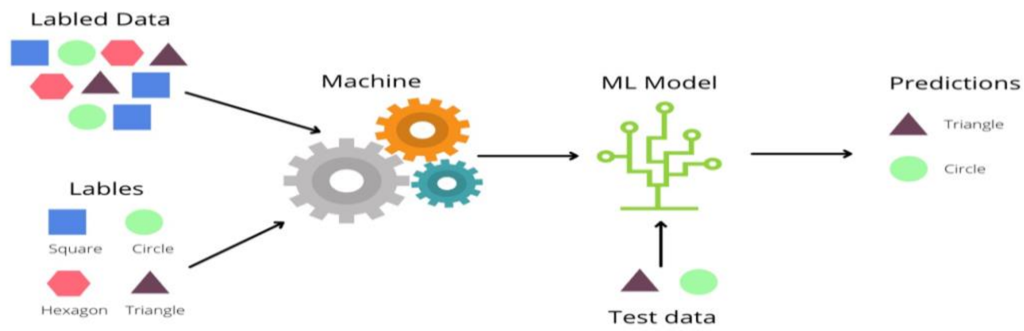


Figure 3 Supervised Learning

Types of Supervised Learning models:

Regression: Regression algorithms come into play when we can see a clear relationship between the input variable and the prediction—for example, weather forecast and market trends, and the prediction variable is a continuous variable.

- Linear Regression
- Regression Trees
- Non-Linear Regression
- Bayesian Linear Regression
- Polynomial Regression

Classification: Classification algorithms come into play when there are two or more classes, and the prediction variable is categorical [11].

- Random Forest
- Support Vector Machine
- Decision Tree
- Logistic Regression

2.2.2 Unsupervised Learning

It is a method where the model is not being taught or supervised. On the contrary, it deals with untagged data. It enables the model to function independently to identify previously unnoticed patterns and information [12]. It is used to find anomalies and unknown patterns in the data.

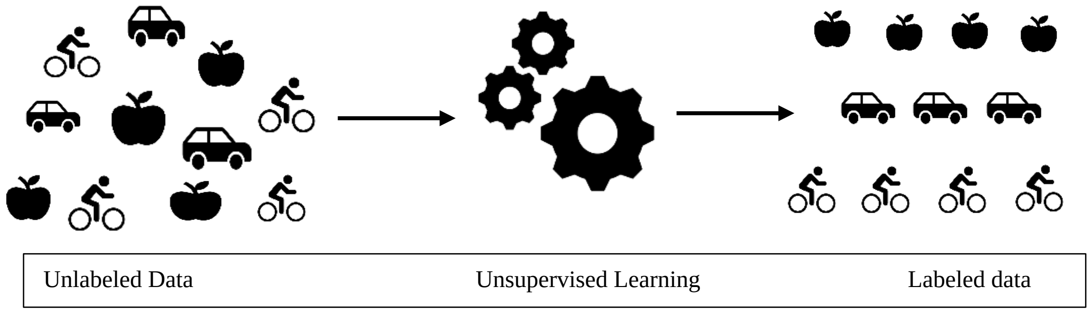


Figure 4 Unsupervised Learning

Types of Unsupervised Learning models:

<u>Clustering:</u> It is used to find the clusters or patterns in an unlabeled dataset. Learning Without Supervision Clustering algorithms will analyze data and look for natural clusters. Also, choose the number of clusters that the algorithms should identify. It lets us change the granularity of these clustering [12].

<u>Association</u>: These rules enable the creation of associations between data elements in massive databases. The goal is to find interesting correlations between variables in massive databases.

Few Unsupervised learning algorithms:

- One Class SVM.
- K-means clustering.
- Isolation Forest.

2.3 Deep Learning Architecture

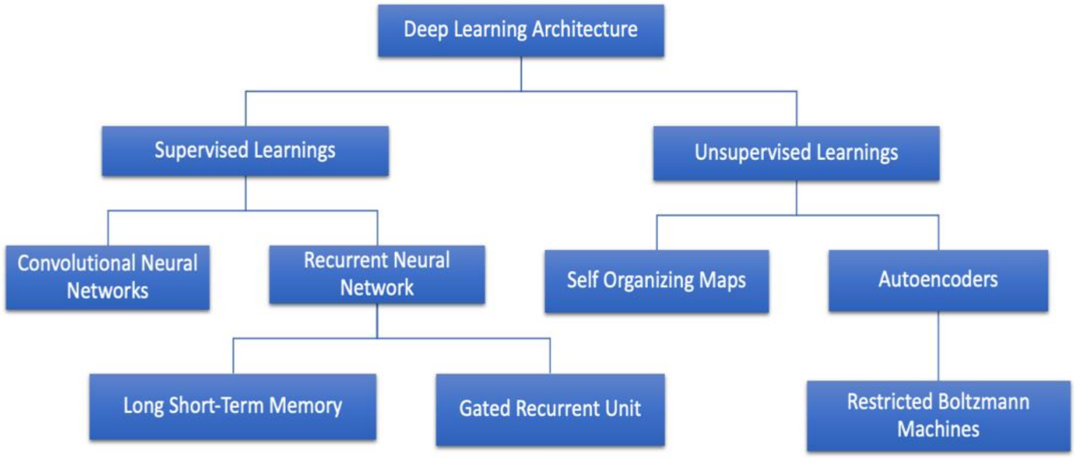


Figure 5 Deep Learning Hierarchy[61]

This section goes into detail regarding deep learning architecture and how they operate to gain a better knowledge of Autonomous Systems. The artificial neural network (ANN) is the fundamental architecture of deep learning[13]. ANN-based algorithms have been used to generate several algorithms. It is comparable to a machine learning framework. It allows us to make better practical use of the technology, speed up work, and support a variety of endeavours without the need to design an ML technique from the ground up. These networks provide the basis of deep learning architectures [14].

A deep learning model comprises many hidden layers. Thus we call such models deep learning models. Deep neural networks are classified depending on how information flows through them. A feedforward-DNN network is one in which information goes from an input layer to an output layer without any feedback connections. One of the functions of DNN is the ability to learn and extract the features from a raw dataset. As the model moves from the lower to the higher end of layers, the features extracted become more pronounced, allowing the learning model to infer accurate solutions for the given prediction or classification task. A representation or feature learning is a neural network's ability to autonomously identify the data necessary for feature detection and classification. Therefore, we used this ability to work to create autonomous systems.

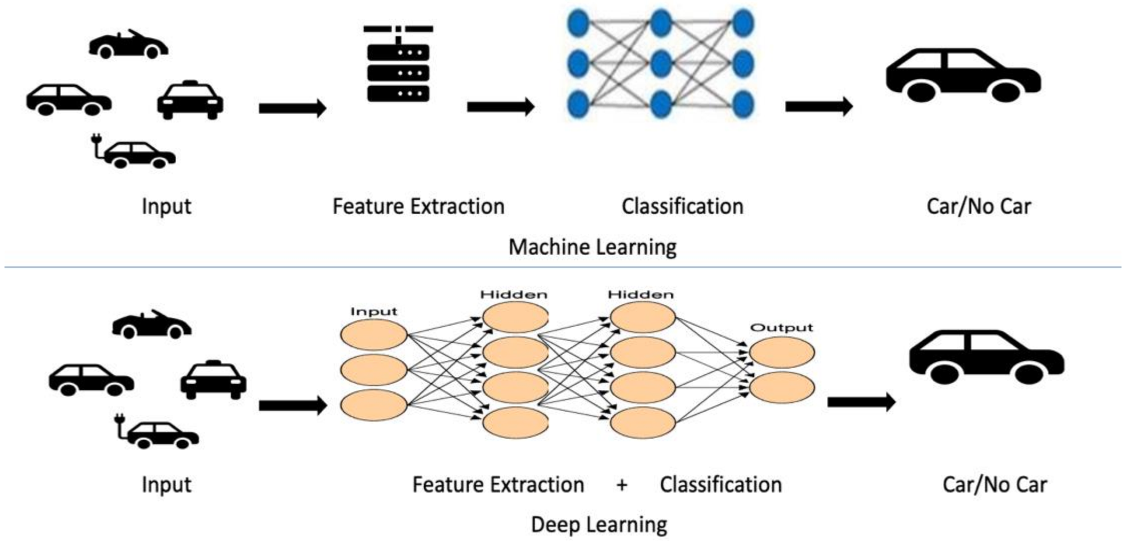


Figure 6 Comparison between Machine Learning and Deep Learning [67]

Fig 6 clearly explains the difference between machine learning and deep learning algorithms. Deep learning algorithms can do classification and feature extraction and predict the output of structured and unstructured datasets as these networks may be used to replace manual picking of domain-specific characteristics, which is essential for many data mining and machine learning approaches. Deep learning algorithms are fast and complex to work on ever-growing and abstract data; it is solid due to their ability to analyze many features. Deep Learning plays a vital role in working on unstructured data, where the comprehension of images/ videos, texts, and audio is known as Computer Vision. As mentioned above in Figure 5, Deep Learning is further worked on Supervised and Unsupervised learning, with convolutional neural networks, recurrent neural networks, and autoencoders.

2.3.1 Convolutional Neural Network

Like regular Neural Networks, Convolutional Neural Networks are composed of neurons with trainable weights and biases. Each neuron gets an input, conducts a dot product, and, if desired, follows it with non-linearity. The main difference is that Conv-Net designs presume that the inputs are pictures, allowing us to embed specific attributes into the architecture. Therefore, implementing the forward function becomes more efficient, and the number of parameters in the network is considerably decreased [15][17]. CNN is a feed-forward deep learning network used to solve visual and text-based issues. Hubel and Wiesel's significant contributions shaped the construction of a CNN [16].

Convolutional neural networks differ from other neural networks because they function better with picture, voice, or audio signal inputs. They have three significant sorts of strata, which are as follows [16]:

- Convolution Layer
- Pooling Layer
- FC (fully connected) layer

<u>Convolution Layer</u>: The convolutional layer is the most significant component of a CNN since it is where most of the computation occurs. It needs input data, a filter, and a feature map, among other things.

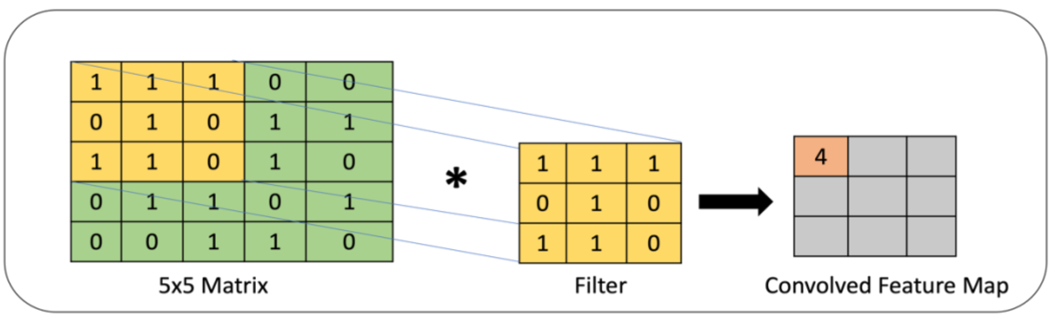


Figure 7 Feature Map calculation using Convolution Layer [16]

<u>Pooling Layer:</u> Down-sampling, also known as pooling layers, is a dimensionality reduction approach that minimizes the number of input parameters. Like the convolutional layer, the pooling operation sweeps a filter across the whole input, but this filter has no weights. Instead, the kernel employs an aggregation function to populate the output array from the receptive field values. There are two types of pooling:

Max Pooling: The filter chooses the pixel with the most significant value to transmit to the output array as it goes through the input. This method is utilized more frequently than typical pooling.

Average pooling: The filter computes the average value within the receptive field and sends it to the output array as it passes over the input.

While the pooling layer loses much information, it has a few benefits for CNN. They help reduce complexity, improve efficiency, and lower the danger of overfitting.

Fully Connected Layer: In partially connected layers, the pixel values of the input pictures are not directly linked to the output layer. Each node in the output layer directly connects to a node in the preceding layer. This layer performs categorization tasks based on the characteristics collected by the initial layers and their filters. While convolutional and pooling layers employ ReLu functions to classify inputs, FC layers use a SoftMax activation function to produce the probability between 0 and 1.

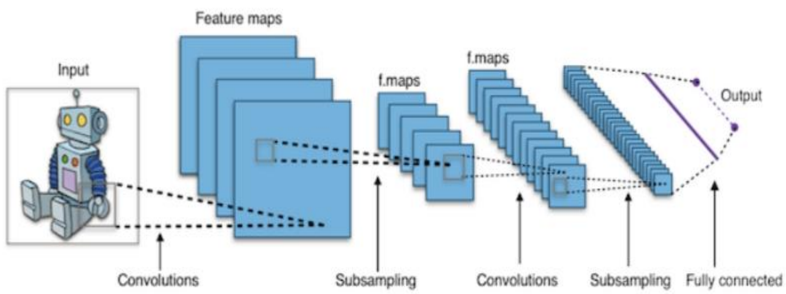


Figure 8 Convolutional Neural Networks [17]

2.4 Region-based Convolutional Neural Network

The goal of RCNN was to take an input image and generate a collection of bounding boxes, each of which included an item as well as the category of the object. RCNN is a recently developed computer vision algorithm that can handle a wide range of computer vision applications. The parts that follow go through some of the RCNN versions that have been created. RCNN begins by using the Selective Search method to extract Regions of Interest (ROI) from an input picture, where each ROI is a rectangle that may detect the border of an item in the image. Depending on conditions, there might be up to two thousand ROIs. The ROIs are then loaded into a neural network, which provides output features [18]. A series of support vector machine classifiers are employed to detect what sort of item is contained inside each ROI's output characteristics. RCNNs are used to localize the objects within an image. The RCNN architecture is explained in Figure 9

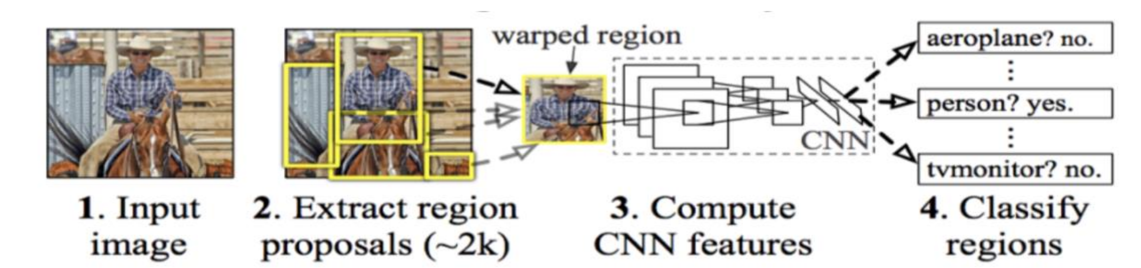


Figure 9 Region-based CNN [18]

Tasks performed by RCNN[18]:

- Selective Search: Before transferring a photo over a network, we must extract area recommendations or regions of interest using a selective search technique. The collected crops must next be shrunk (wrapped) and distributed over a network.

- Extracting Region Proposal: Selective Search is a region recommendation approach for object localization that mixes regions depending on their pixel intensities. Therefore, pixels are organized in a hierarchical grouping of similar pixels.

- Positive vs Negative example: After the area of suggestion is extracted, labelling the area for training is done. As a result, classifying all proposals with Intersection over

Unions(IOUs further discussed in subsection 4.1.4) of at least 0.5 that have any of the ground-truth bounding boxes are foreground or positive examples. On the other hand, all other area suggestions with an IOU of less than 0.3 are labelled as background or negative examples. As a result, the remainder of them is just disregarded.

- Bounding box regression: Object identification techniques frequently use bounding-box regression to improve or anticipate localization boxes. The equations below depict the target that CNN will be looking for. As a result, the center coordinates are x and y, where w and h stand for width and height, respectively for the target object. Finally, G and P are abbreviations for the ground-truth bounding box and region proposal. It should be noted that the bounding box loss is only evaluated for positive data.

$$t_x = (G_x - Px)/P_w$$

$$t_y = (G_y - P_y)/P_h$$

$$t_w = \log(G_w/P_w)$$

$$t_h = \log(G_h/P_h)$$

- Loss: The overall loss is calculated by adding the classification and regression losses. The latter, however, has a coefficient lambda of 1,000 in the original article. The regression loss is neglected under adverse situations.

2.5 Faster-RCNN

Faster RCNN is an extension of RCNN. RCNN and Fast-RCNN both work on the selective algorithm, but to be more advanced, the algorithm is rather sluggish and time-consuming. Faster RCNN, on the other hand, includes an object identification method and the network's ability to learn the region of suggestions. This sends images through the convolutional network to produce convolutional feature maps. Further, the predicted region proposals are reshaped using an ROI pooling layer, which can categorize the image inside the suggested regions and forecast the label values for the bounding boxes [19].

Faster-RCNN has two stages [19]:

- The first stage is to construct a Region Proposal Network, which is a deep fully convolutional network that generates region ideas (RPN). The RPN module is in charge of directing the unified network's attention.
- In the second phase, the Fast RCNN detector is utilized, which harvests features from each candidate box using ROI-Pool and performs classification and bounding-box regression.

2.5.1 Region Proposal Network

The Region Proposal Network accepts any size picture as input and creates a sequence of rectangular item ideas, each with its objectness score. Objectness is a measurement of the model's level of certainty that the anchor box contains an item. On a computer, this is performed by dragging a tiny network across the feature map generated by the convolutional layer RPN in combination with a Fast-RCNN object recognition network. RPN features are fed into two linked sister layers: a box-regression layer for bounding box regression and a box-classification layer for object classification. RPN is quick, creating ROIs for each image in less than 10 milliseconds.

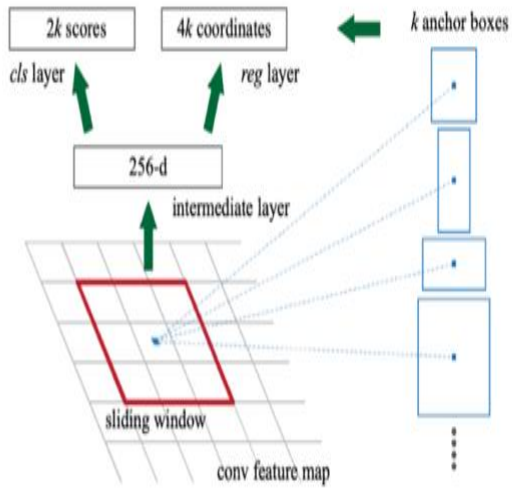


Figure 10 Region Proposal Network [19]

2.5.2 Anchors

An anchor is related to a scale and aspect ratio and is situated in the center of the sliding window. RCNN generates nine anchors for each sliding window using three scales and three aspect ratios. These aid in the prevention of translational inconsistencies. It tries to predict multiple region proposals for each sliding window position simultaneously. The letter k indicates the maximum number of ideas that can be made for each location. The regressor layer gives 4k outputs that encode the positions of k boxes, whereas the cls (classification) layer generates 2k scores that evaluate the likelihood of each proposition being object or non-object [19].

2.5.3 Architecture of Faster-RCNN

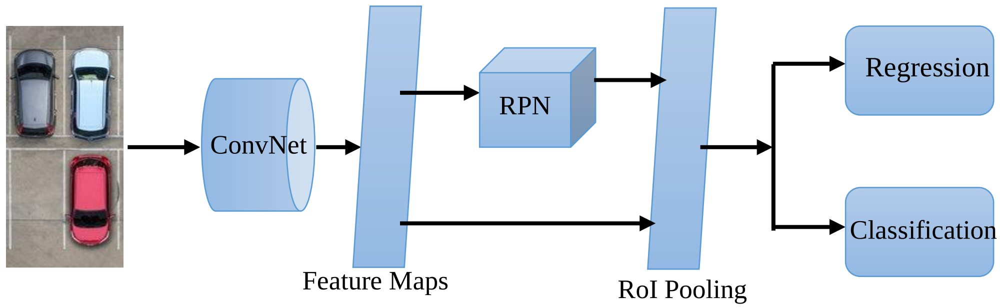


Figure 11 Overview of Faster-RCNN

RPN is a network that generates region proposals, Whereas Fast-RCNN is used for detecting the objects within the regions proposed. These two components together make a Faster-RCNN. As an overview of the whole network, the image is sent through a convolutional network, which produces a convolutional feature map. Rather than using a selective search technique on the feature map to generate the region recommendations, a separate network is used to anticipate the area suggestions. An ROI pooling layer is then used to reshape the predicted region proposals, which is eventually used to classify the image inside the proposed region and predict the offset values for the bounding boxes [19].

2.6 Mask-RCNN

2.6.1 Mask-RCNN Architecture

Mask-RCNN is a neural network built on top of Fast-RCNN and Faster-RCNN. It is one of the examples of a segmentation framework that recognizes objects while simultaneously providing segmentation masks for every occurrence. It is known as the new state-of-the-art, Instance Segmentation. Image Segmentation is the process of dividing the image into multiple image segments, known as image objects or regions. Therefore, it can differentiate between numerous classes within an image or video with returning classes, bounding boxes and masks of that particular class.

2.6.2 Backbone

Mask-RCNN uses a convolutional backbone architecture to derive feature mappings from an input image. Feature Pyramid Network(FPN) is not to be confused with a featured image pyramid, which employs multi-scale images to calculate feature maps at each size; instead, FPN takes a single-scale image as input and outputs feature maps at many sizes. The FPN has three pathways: bottom-up, top-down, and lateral. The bottom-up route is the feedforward computation of the convolutional backbone, which divides the CNN into stages based on each layer's output size and samples each stage's final output as the feature map of that scale to form a hierarchical feature pyramid. Conv2 through conv5 are utilized in ResNet with respective strides of {4,8,16,32} pixels, equal to a scaling step of 2, to generate feature maps C2, C3, C4, and C5. ResNet50 and ResNet101 backbones have been used for this research [19].

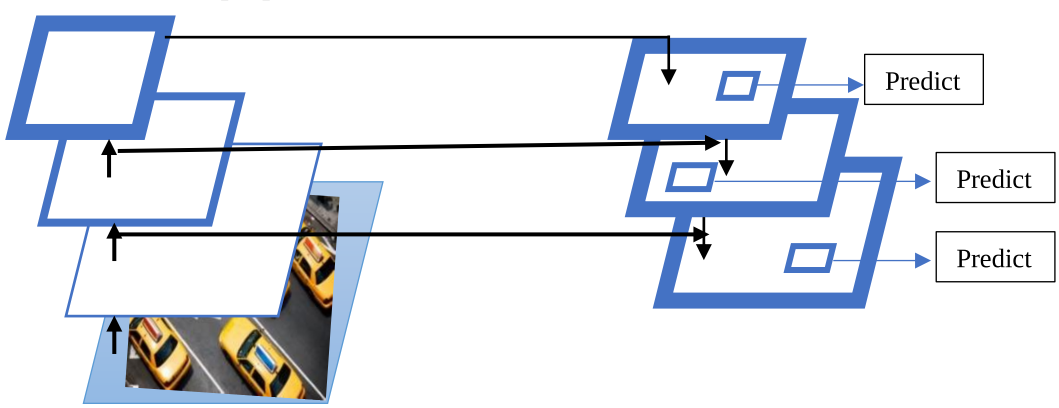


Figure 12 FPN architecture. Blue line thickness indicates semantic strength [66]

In the top-down technique, the semantically more robust high-level feature maps are interpolated to generate higher resolution low-level features, with a scaling step of 2. They are strengthened by lateral connections, 1*1 convolutional layer, and matching levels in the bottom-up feature pyramid. The pathway begins by performing a 1*1 convolution on the coarsest feature map in the bottom-up pathway, and the interpolation is repeated until a feature pyramid P2, P3, P4, and P5 are formed [20].

2.6.3 Region Proposal Network

Region Proposal Networks are the same as Faster-RCNN, Mask-RCNN uses the same framework as RPN.

Regions of interest (ROIs), or candidate bounding boxes, can then be identified on each level of the pyramid. to generate ROIs, Mask-RCNN uses the region proposal network (RPN) introduced in Faster-RCNN.

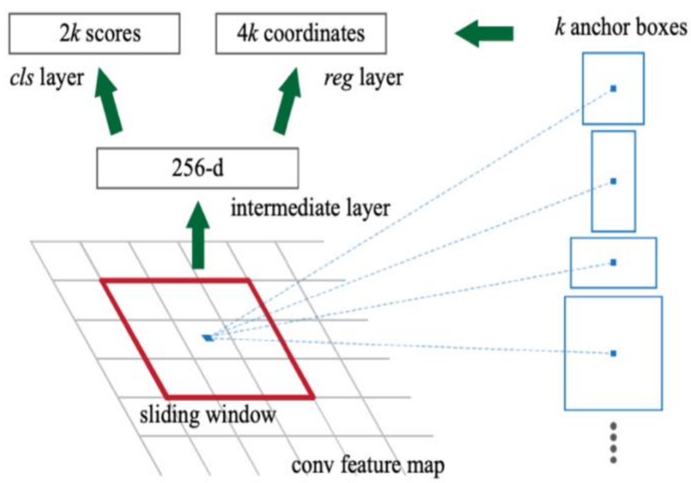


Figure 13 Region Proposal Network [19]

2.6.4 Region Of Interest (ROI) Align

ROI Align, or Region of Interest Alignment, is a method used in detection and segmentation applications to generate a little feature map from each ROI. It reduces the severity of the ROI Pool quantization, allowing the extracted features to be aligned with

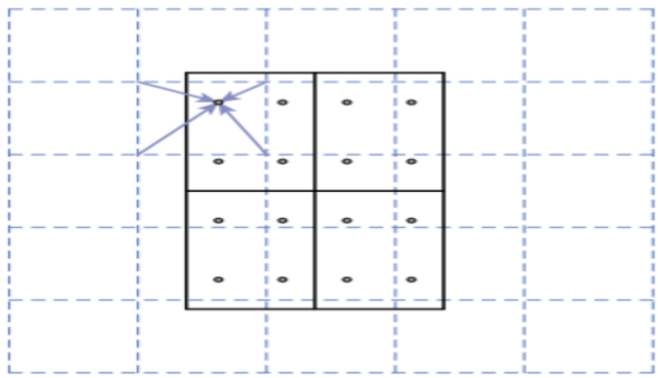


Figure 14 ROI Align [18]

the input. Instead of rounding ROI boundaries to feature map bins, ROI Align uses the feature map to bilinearly interpolate four sample points in each ROI bin as shown in Figure14.

2.6.5 Network Head

The network head performs classification, bounding-box regression, and mask predictions on each ROI. It employs the same ResNet and FPN architecture as the backbone but adds a mask-predicting CNN branch. The network head extends the ResNet and FPN designs with a convolutional branch. Layers such as conv (3x3), deconv (2x2 with stride 2), and fully connected (fc) are offered. Padding allows conv layers to retain 2D resolution, deconv layers to improve it, and fc layers to keep 3D resolution [20].

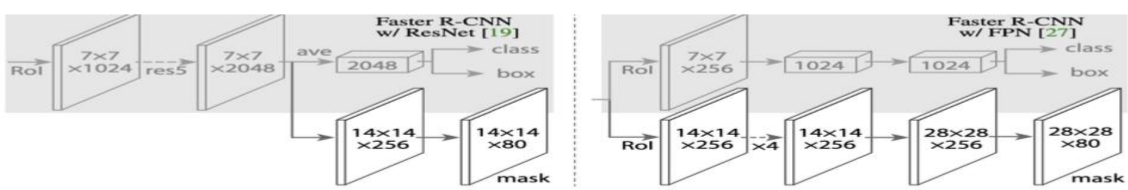


Figure 15 Network Head [20]

2.6.6 Comparison between Mask-RCNN and Faster-RCNN

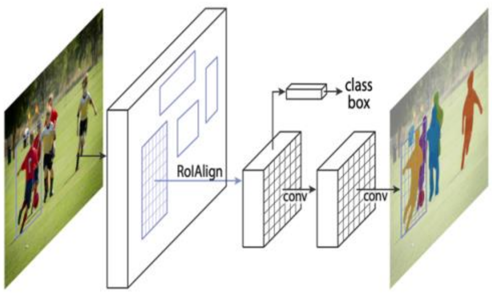
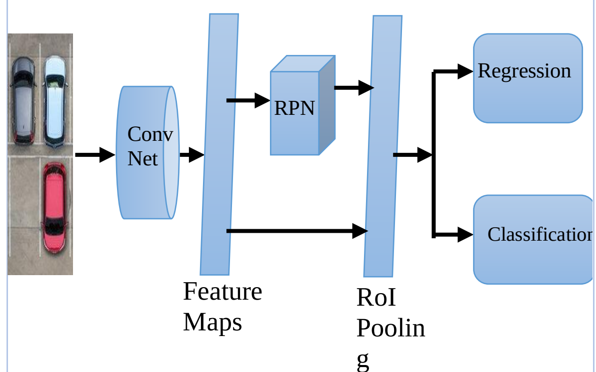
Mask-RCNN	**Faster-RCNN**
Works on semantic Segmentation + object detection	Works on object detection only
Gives label and bounding boxes as output	Does not give bounding box + label outputs.
Creates mask of the object	NO mask
Works on top of Faster-RCNN	Works on top of CNN and Fast-RCN
 [20]	

Table 3 Comparison between Mask-RCNN and Faster-RCNN

2.7 Fundamentals of Image Pre-Processing

Computer vision is a branch of artificial intelligence (AI) that allows systems to extract meaningful information from digital photos, videos, and other visual inputs and then act or make suggestions based on that knowledge. Computer vision teaches computers to execute human-like duties, but they must do so in a much shorter time using cameras, data, and algorithms. Computer vision needs a massive quantity of data. It analyses data repeatedly until it identifies differences and eventually recognizes images. To teach a computer to identify car tires, for example, enormous numbers of tire photographs and tire-related documents must be fed into it to learn the distinctions and distinguish a tire, particularly one without flaws [21]. Two technologies are used for this task: deep learning, machine learning, and convolutional neural network (CNN).

Machine learning uses computational models to educate a machine in the context of visual inputs. If sufficient data is fed into the model, the computer will "examine" the data and learn to distinguish between pictures. Algorithms enable a computer to remember on its own rather than being taught to recognize a picture. CNN segments images into tagged or labelled pixels. It uses the labels to do convolutions and anticipate what it observes. The neural network runs convolutions and evaluates the accuracy of its predictions in iterations until the predictions begin to come true. It then identifies or perceives images in a way that humans do. Convolutional Neural Network is widely used in computer vision applications such as picture classification, object identification, image segmentation, and many more.

Image classification is one of the fundamental approaches in today's world; it is used in various sectors such as healthcare, business, and many others; therefore, learning and developing state-of-the-art computer vision models is essential if working in artificial intelligence.

2.7.1 Image Processing

By the 1960s, many approaches for processing digital pictures were under development. They were employed at numerous extreme locations, including the Jet Propulsion Laboratory, MIT, Maryland University, Bell Labs, and several other research sites and organizations. The technology is capable of, but not limited to, medical imaging and satellite photography, among many other applications. Initially, the goal of image processing was to increase image quality. Processing might improve the quality of the image or extract meaningful information from it. It has uses in areas like medical imaging and may even be used to disguise data inside an image.

In image processing, a low-quality image is an input, whereas the output is an image with much-increased quality. Picture processing can include many different features, such as image improvement, restoration, encoding, and compression. People frequently confuse image processing with computer vision. It is a subset of Computer Vision, although the two are not synonymous. Image Processing systems focus on converting pictures from one format to another, whereas Computer Vision systems assist the computer in understanding and extracting meaning from images. Many modern image processing approaches use Machine Learning Models, such as Deep Neural Networks, to modify pictures for a range of objectives, such as adding creative filters, tweaking an image for optimal quality, or improving certain image features to maximize quality for computer vision tasks. Recent machine learning approaches enable engineers to enrich image data in addition to doing image manipulation.

2.7.2 Image Augmentation

Image augmentation is a method of changing existing data to produce unique data for model training. Therefore, it is a method of artificially expanding the available dataset for training a deep learning model. Different augmenting techniques create new data or preprocess the data [22].

Image Rotation: This is a prevalent task to perform, where we are just rotating the image. The object within the image remains the same and recognizable but is used to enhance and increase the training data.

Image Shifting: Image shift is a geometric transformation that converts the position of each object in an image to the new location of the final output image.

Image Flipping: Flipping is an extension of rotation, and it allows us to flip the picture in both the Left-Right and Up-Down directions.

Image Noising: It is the process of adding noise to a picture. This strategy teaches our model to distinguish between the signal and the noise in a picture. This also makes our model more resistant to image alterations.

Image Blurring: Images are gathered from several sources, and as a result, the quality of each source varies. Every image obtained is of different qualities. In such circumstances, we may blur the source photographs, making our model more resistant to the image quality used in the test data.

2.7.3 Image Enhancement Techniques

Contrast enhancement is a critical component of image processing for both human and machine vision. This is commonly employed in medical image processing and a preprocessing phase in voice recognition, consistency synthesis, and a variety of other image and video digestion applications. For this objective, various approaches have been devised.

Histogram equalization (HE): It is a general approach for enhancing image contrast. Because of its simplicity and much-improved performance on virtually all sorts of photos, it is the most often used approach. HE works by remapping the grey levels of a picture based on the probability distribution of the input grey levels. The two forms of histogram equalization are global and local histogram equalization [23].

- Global Histogram Equalization (GHE) is a transformation function that uses the histogram information from the whole input image. In contrast, this global approach is appropriate for overall enhancement.

- Local histogram equalization (LHE) uses a small window that slides through every pixel of the image sequentially, and only the block of pixels that fall within this window are considered for HE, and grey level mapping for enhancement is done only for the centre pixel of that window [23].

Another approach is to employ non-overlapping block-based HE. Nonetheless, these techniques typically produce unappealing checkerboard patterns on enhanced pictures.

Dynamic Histogram Equalization(DHE): It is the method where we lessen the dominance of higher histogram components over lower histogram components in the image's histogram and control gray level stretching for optimal image feature enhancement. DHE applies the transformation function to the whole histogram at once but splits it into sub-histograms until there is no dominating region. Each sub-gray histogram's values are mapped to a dynamic gray level (GL) range by HE. This is accomplished by distributing the available dynamic range of gray levels among sub-histograms according to the dynamic range of the input image and the cumulative distribution function (CDF) of the histogram values. This range of contrast stretching prevents the obliteration and bleaching of tiny image features and ensures a fair contrast boost for the whole image. For each sub-histogram, a unique HE transformation function is produced, and the gray levels of the input image are transferred to the output image [23].

Contrast-limited adaptive histogram equalization (CLAHE): It is an adaptation of adaptive histogram equalization in which the contrast amplification is reduced to solve the issue of noise amplification. The angle at which the transformation function is applied in CLAHE is what establishes the level of contrast enhancement that is applied in close proximity to a particular pixel value [24].

Fusion Exposure Framework: It works as the foundation for the exposure fusion and contrast enhancement strategy. Using illumination estimation approaches, a weight matrix for image fusion is generated. Then, how the camera response model generates photos with many exposures is illustrated. The optimal exposure ratio is then determined such that regions of the synthetic picture that were underexposed in the original image are now well-exposed. In accordance with the weight matrix, the input picture and the synthetic image are then combined to produce the enhanced image. Experiments demonstrate that method produces less contrast and brightness distortion than previous methods [25].

Chapter 3

3 Literature Review

Decades of research have been devoted to examining parking lot detection. There were few early models based on core Machine Learning ideas and human-managed systems at the outset of research into autonomous parking lot systems. Even if the technology were to be deployed, the applications were not practical and expensive. There are three main methods for designing an intelligent parking system [26]:

- Counter-based: These systems are the most traditional ones that we use at almost every paid car park, a strictly gated system. Cars are being counted while entering or leaving the facility. Human efforts are needed to detect which parking is vacant and which is not for spot detection.
- Sensor-based: These kinds of a system depend on the functioning of the sensors deployed at the parking lot. Usually, it tracks the movement and location of the car, whether within the slot range or not. Once the occupancy is detected, the system returns the output of occupancy details to the user. These are expensive to install, have high operational costs for maintenance, weather dependent for outside parking lots, but they give satisfactory results.
- Image or Vision-based: These systems reply to the image/ video feed and then try to extract the details about the parking [26].

Finding a parking spot has become the most rigorous task in our daily lives for quite a while. The sensor system, therefore, marked the beginning of intelligent parking. Using the internet of things, the author [26] constructs an automatic parking system that allows the user to locate the nearest accessible parking place. The architecture of the parking system consists of many components, including a centralized server, a Raspberry Pi, an image capture system, a navigation system, a display device, and a user device. Basavaraju et al. [27] demonstrate that the cloud-based IoT design incorporates a cloud service for storing information about parking spot availability and a centralized server for storing smart parking system data. Using infrared sensors, RFID(Radio Frequency Identification), and IoT. Another author [28] creates a prototype for an IoT-based car parking management system for smart cities. This prototype includes parking spot availability, IoT maintenance of a parked car database via a shared server, slot booking, and theft management. Additionally, it offers several benefits, including reduced human interaction, increased flexibility, and enhanced security. This system includes online reservations, parking entry, exit, and parking management modules.

Patil et al. [29] describe a concept that might monitor and manage cars in a parking garage by informing drivers of available parking spots and directing them to the appropriate area. The technique involves modifying the original WSN(Wireless Sensor Network) and using RFID(Radio Frequency Identification) and ZigBee (standards based wireless technology) technologies, and time and other pertinent parameters should be examined. Each sensor node collects data that is handled in either a distributed or centralized manner.

There are various alternative approaches where systems provide end-to-end parking through sensors, similar to valet parking. Bluetooth is a wireless technology standard for short-range data transmission that is utilized as a means. Autonomous system technology uses a mechanical device to transfer vehicles between parking lots without driver intervention. As the driver approaches the parking area, he places the vehicle on a movable platform and activates his Bluetooth to initiate the parking procedure. The Bluetooth reader maintains the user's ID in a database, and the ARM microprocessor verifies the ID against the saved digits. The parking process begins when a new vehicle is obtained; when an existing vehicle must be retrieved, the retrieval process begins [30]. These systems are expensive. Therefore, a system is introduced that provides a wireless method for remotely recognizing parking spots using a smartphone. This strategy makes parking easier to locate and pay for. If a vehicle is detected in a parking space, no action is taken; however, if the parking spot is vacant, the wireless sensor node broadcasts the location to the servers for each parking area. When a user is within 2 kilometres of a parking spot, the server locates and alerts the user of the location of the closest accessible parking space [31].

An autonomous smart vehicle parking based on a genetic algorithm demonstrated a self-determining streetcar framework and a method provided by D. Thomas et al. [32] for robotizing autos in a shopping mall. The self-governing streetcar framework and the genetic algorithm are applied to determine the optimal stopping opportunity, hence making the proposed framework efficient in terms of space utilization and streetcar efficacy. The genetic technique made a difference, as their proposed framework made progress in sifting interaction for the idealized location without wasting time. The Free streetcar framework reduces clients' waiting time in line. The client may exit their vehicle within the entrance, and the automated streetcar will stop with care. This hypothesis is used to calculate the utilization, productivity, and holding time of the framework. Compared to a framework without a genetic algorithm, the introduction demand chart demonstrates that a Genetic algorithm increases utilization and capacity without sacrificing the reduced holding length. The genetic computation method and application of the lining hypothesis have assisted in mitigating the problem of parking lot scarcity and vehicle overbooking [32].

Table 3 below provides sensor-based Parking Detection methods and a summary about how the authors tried to create a system to solve our parking-related issues.

Table 4 Sensor based systems

Reference	**Summary**	**Sensors+ Deployment**	**Disadvantages**
Patil, M., & Bhonge, V. N. et al. [29]	RFID is utilized for vehicle check-in and check-out that is quick, safe, and easy. On the other hand, ZigBee Wireless Technology allows for secure data exchange. The system checks to see if the car arriving for parking is registered, and the data is saved in a database. The parking area's LCD displays give information about available parking	RFID , Zigbee	The technology allows for one-on-one parking but is time intensive and limits the number of concurrent check-ins and check-outs. The driver is unable to reserve a parking place using a mobile device. -The usage of RFID raises the cost.

	spaces and direct vehicles toward parking.		
Karbab, E & al. [33]	The LIBP collection protocol is used to manage energy in a heterogeneous wireless sensor network. Drivers may make use of a variety of CPF services. In terms of energy usage and financial cost deployment, the CPF distinguishes two performance groups.	Ultrasonic sensors + RFID, Zigbee	Vulnerable from theft point of view.
Gandhi et al. [28]	The user may search for available parking spots. IoT allows for data storage, processing, and collection. Where drivers can reserve slots. The handling of theft has been accomplished. There is less interaction between people. Increase in flexibility and security.	Infrared sensor, Zigbee, RFID	Not valid for illegal parking slot detection.
Orrie, O. et al. [31]	Wireless sensors being used, gives the accurate availability of the parking lot. Smartphones connected for more feasibility. Also, GPS gives an accurate location to the user for parking.	Sensor+RFID	Smartphone location tracking valid up to 2 kilometers

Sensor-based methods are usually very efficient, but the drawbacks of these systems are being very high maintenance, overpriced and do not work well with all weather conditions.

Therefore, vision-based systems are being used to create cheaper and more reliable systems. Al-Kharusi et al. [34] discuss the system, which takes in input as images and works on them by defining a spot at the center of every slot to define the parking area; the image is then further changed into several forms to detect the accurate results for parking occupancy. Steps like Image acquisition and separation are done. HSV conversion is done to process these.

These parameters and results were imprecise and used very traditional techniques, but automation needed to be done. Methods like Haar Cascade and Histogram of Oriented Gradient (HOG) came into play. Histogram of Oriented Gradient is used for object detection in parking lot detection. The system is fed with occupied and unoccupied labels into the SVM classifier. In these methods, the HOG is used for feature extraction and produces results ranging from 91% to 94% [35]. Whereas using Haar Cascade with Adaboost classifier is reviewed by Fusek et al. [36], the system is trained on the images of a fish-eye view, where the parking slots are hardcoded so the classifier can easily predict the occupancy. While there are many works using these algorithms, but they are still time-consuming and less accurate and involve predefining the spaces.

Reviewing several existing methods revealed that deep learning algorithms could present better and fast results. In the deep learning literature, the parking classifications are done using two approaches: (i) Space-based- where the system is being trained on the parking lots labelled as occupant and vacant, at the end the detection is done on space and not a car. (ii) Car-based- The detections are being done separately on a car where the algorithms are being fed with images annotated with the car class. The slot detection is done separately for this [26].

In deep learning, the first algorithm used is Convolutional Neural Network, which can be used for high computational systems. Batch normalization is applied to input layers for enhanced performance and results. Nurullayev and Lee [37] proposed a Convolutional Neural Network (CNN) approach based on dilated convolutions that yielded promising results by removing pixels in the convolution kernel, enhancing the classifier's learning capacity in the pictures' global context. These were only a handful of CNN theories. Several modifications and recreations may be done, resulting in a wide range of discoveries and research. With diverse architectures and systems, CNN is the driving force underlying vision-based learning.

Acharya et al. [38] suggested a CNN-based real-time parking recognition system that employed a pre-trained CNN model for feature extraction and an SVM classifier to find the most occupied parking spaces. This methodology relied on traditional inference methods while using DL as a training module. The technique achieved 99.7 percent accuracy on public datasets and 96.6 percent accuracy on bespoke datasets. Another research implies that SVM might be replaced by other methods that provide faster and more consistent outcomes.

Mora et al.[39] and Dhuri et al. [40] used the original VGG16 network in their approaches. Mora et al.[39] evaluated their model using the PKLot dataset, which considered camera angles, ambient conditions, and parking lot variances. Furthermore, the authors suggest a method based on a Bag of Features. Whereas Valipour et al. [41] built a VGGNet-F model from the ImageNet dataset and then fine-tuned it with PKLot images, resulting in improved generalization capabilities.

G.Amato et al. [42] extend the work on this concept by adopting two unique CNN architectures, mAlexnet and LeNet. These models' accuracy did not suffer, but they did take longer to compute. Based on the core idea of these models, J. Nyambal et al. [43] worked on them using Caffe and Nvidia. DiGITS indicated that the system had achieved 99 percent accuracy, whereas LeNet still surpasses mAlexnet by a small margin. Rahman et al. [44] employed mAlexNet in a recent study. However, the kernel size of the first layer was changed. There was no discernible difference in the outcomes.

The research for smart parking has been extended to find better versions of deep learning techniques to detect more reliable and efficient solutions. Region-based algorithms were brought into the picture to work more deeply on detecting cars and parking lots. Faster-RCNN is another example that is being under work. These provide more structured results, and Faster-RCNN has improved the quality of Regions of Interest proposed by the Region Proposal Network. CNN model is used as the backbone in these techniques to achieve feature extraction. On the other hand, using Faster-RCNN on a parking system may be beneficial. Martin et al. [45] used this model to identify vehicles, using homographic transformation and correction to change the plane of the camera receiving pictures to a common plane. Finally, the automobiles recognized by cameras were fused to locate the appropriate parking places.

Poply et al. [46] proposed a system based on Faster-RCNN to provide a decentralized approach for parking detection system using real-time detection. Detection of spots is done while using stationary car parking detection. This was the example of parking spot occupancy detection using the stationary parked cars as the spots get detected, and then once reviewed after this, it can be able to detect if it is vacant or not.

Another CNN extension is Mask-RCNN. It is used in processes such as segmentation. It outperforms the others in terms of computing efficiency and descriptiveness. Mask-RCNN, as the name suggests, generates a mask for the region. It produces labels and boundary boxes while showing the findings. Mask-RCNN has been employed in various research since its debut because it generates more accurate results and is more robust and easier to use.

J. Ahmad et al. [7] worked on Mask RCNN to create a parking system. The author used a rudimentary model to receive and investigate the procedure with COCO mode. Naufal et al. [47] worked on preprocess Mask RCNN where the author preprocessed the pictures with image enhancing techniques Exposure Fusion framework to minimize noisy disturbances and supplemented further to have more data. The parking lots and their occupancy were also discovered. After that, mAlexNet was utilized to determine if the occupancy findings were correct.

Sairam et al. [8] suggested a technique based on the Mask-RCNN network (2020). It was used to differentiate automobiles from two-wheel vehicles by extracting individual vehicles and identifying the fraction of the parking space they occupied. Xu.C et al. [48] created a system where the Resnet-86 network was built and used as a backbone network. To optimize network discovery speed, even more, we doubled the number of reserved RPN candidate frames. To increase accuracy, we developed the SF-FPN approach for feature extraction. The authors are actively working on improving the dataset, FPN, RPN, and network.

Table 5 Analyses of deep learning algorithms

Author	Network	accuracy	Dataset
Amato et al. [42]	mAlexNet	99.6	PKLot
Gabzdyl et al. [49]	VGG encoder/Custom CNN	93	PUCPR
Gregor et al. [50]	Custom ResNet	99.9	PKLot
Mettupally & Menon et al. [51]	Mask-RCNN	91.9	PKLot
Agrawal & Urolagin et al. [52]	Mask-RCNN	95	PKLot
Khan et al. [53]	Faster-RCNN	99.9	PKLot
Agrawal et al. [9]	Mask-RCNN	95	PKLot
Siddharth et al. [54]	Mask-RCNN	84.9	PKLot
Nguyen et al. [55]	AlexNet, mAlexNet, mobileNet	99.5-97.3	PKLot, CNR Park
García et al. [56]	Mask-RCNN	97.98	Custom

The work that has previously been done on this topic and on comparable datasets can be seen in the table referenced before. It depicts the architecture of each project that the research team is working on and the accuracy of their findings. Therefore, this explains G.Amato et al. [42], who implement two different CNN architectures, namely mAlexNet and LeNet. The amount of time needed to calculate these models' findings was significantly prolonged, even though they were not changed in any manner that would render them less accurate. G.Khan et al. [53] also used Faster-RCNN, and the author used the core Faster-RCNN methodology while working on a video stream to distinguish the car. Both methods were successful. After that, it looks for available parking places and tries to find them. This model's performance is much better than that of the other approaches. The CNN and Faster-

RCNN approaches, despite their complexity, produced findings with an accuracy of 99.9 percent, like those described above.

Image processing and object identification are two areas that see widespread use of Mask-RCNN and other deep learning methods. These, together with the concept of fine-tuning a pre-trained model, may be used to develop an excellent system. It allows the user to include and construct many approaches to generate the best result possible. Most of the articles focused on parking lot detection with the Mask-RCNN model as their primary tool. They experimented with different setups, epochs, and learning rates, and then trained extensively to identify the effects of those modifications. Sending additional notifications to users with the assistance of the Twilio module [57] regarding the number of open parking spaces. This fundamental concept was developed further by Y.Song et al. [58], who changed the identification of parking spaces into the detection of vehicle position movements. This considerably decreased the amount of computational complexity and produced much improved results.

Approaches based on Mask-RCNN were also built, and although these methods produced respectable results (around 95–98 percent), they still have room for improvement and may be worked on further. A system that was developed by Agrawal et al.[9] that operates on several perspectives, including an aerial view and a low angle to be worked on, was able to provide an accuracy of around 95 percent while having an IOU of 0.2. The system was designed to recognize parking lots from both locations. On the other hand, our suggested system operates on Mask-RCNN, which monitors live video feeds from aerial views to determine whether or not parking spaces are occupied. The system communicates the recorded state of the parking availability to the application, which in turn assists the user in booking the open parking place and subsequently assists the user in navigating towards the booked parking spot. While we analyzed the PKLot dataset, we focused our attention on ResNet101 and ResNet50 to determine which delivered superior results. In addition, we worked on enhanced photos to improve the detection results, receiving mAP around 0.99.

Chapter 4

4 Methodology

In section 4.1, we discuss the datasets and libraries which are utilized in our experiments. In section 4.2, we explain the annotation process which is performed on the datasets. Next in section 4.3, we describe the data preprocessing and augmentation which is required for the model described in section 4.3. Subsection 4.4 discusses the parking detection system in depth, including data training, testing, and deployment. Also included are two sections for Faster-RCNN and Mask-RCNN. Finally, Section 4.5 describes the technique and design of the mobile application for the parking detection system.

4.1 Datasets and Libraries

4.1.1 Dataset

Dataset is a collection of information or data, that can be worked on by machine learning to help create a predicting model. There are several public domain datasets. In this section, we discuss the dataset called PKLot [59] and our custom dataset.

PKLot: This dataset contains a collection of robust images of parking lots. This dataset is used to train or test parking detection systems or similar projects. The images within this dataset are cropped and labeled, and the patches vary in size and aspect ratio. The images were taken in two distinct parking lot under the same weather conditions if compared to another public dataset called CNR-Park. Images were shot at Federal University of Parana in Brazil. The dataset was separated into three sets for training, validation, and testing, having 24832,17382 and 7450 photos, respectively. There are 52.8 percent vacant gaps and 47.2 percent filled places among the 711856 annotations [59].

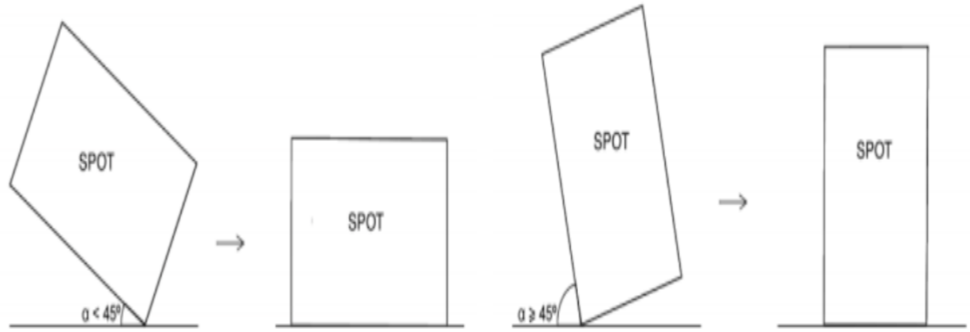


Figure 16 Skew adjustments done in PKLot dataset [59]

The parking spaces in the real parking lot where the images were taken are slanted at 45 degrees, and the authors chose only legitimate parking locations (those indicated by road lines), as seen in Figure 17. The skewed segmented parking spaces were rotated to a 0-degree angle in Figure 16. The sample is also skewed, with 48.54 percent of patches filled and 51.46 percent unoccupied[59].

Figure 17 Original source parking lot for PKLot dataset with skew adjustment example [59]

<u>Custom Dataset:</u> The custom dataset is created by utilizing the Pine tool [64] to generate numerous pictures with image upgrades. The collection features approximately 500 images depicting a variety of weather and noise levels. This was designed to validate the system's output. The web application is used to modify photos for producing and storing our data set in several versions, enabling the system to identify any kind of image from a live video feed. For example: overcast, snowy, and granular.

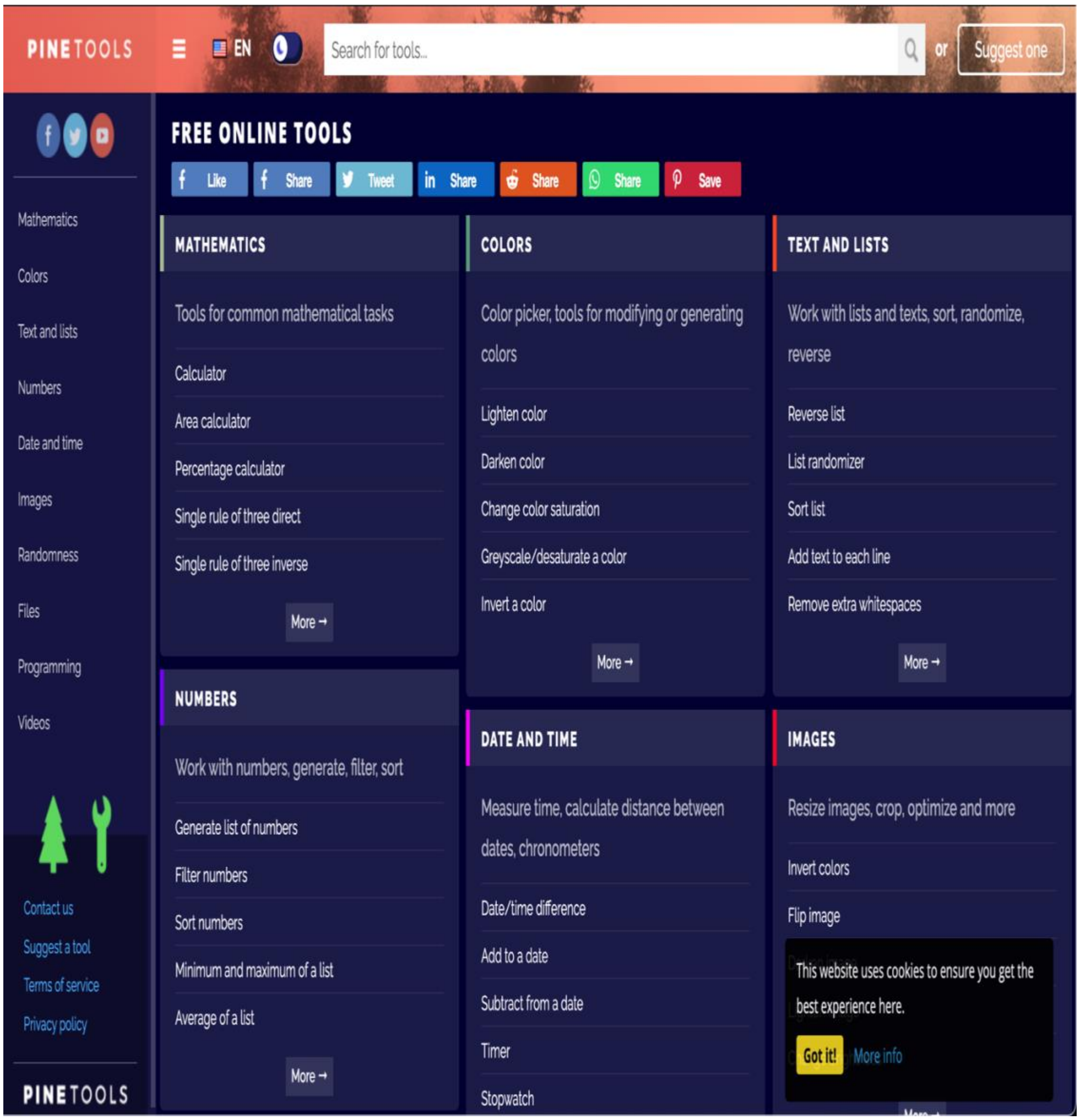


Figure 18 Image Creation using Pine tool

4.1.2 Libraries

Google's TensorFlow is an open-source framework for machine learning. To support a range of computer architectures, the library is built in the programming language C++. TensorFlow applications for machine learning are now supported on mobile devices and websites. Additionally, it is compatible with a broad array of GPUs, CPUs, and TPUs (Tensor Processing Unit). TPU is a specialized ASIC-based hardware component designed for deep learning computations. TensorFlow is among the most effective deep learning frameworks.

Keras is a high-level Python API meant to facilitate rapid prototyping and make the process of experimenting and creating more efficient and user-friendly. The library is based on TensorFlow and other backends. It was recently added in the official release of TensorFlow. Together, they provide the most adaptable foundation for in-depth study.

PyTorch is a GPU and CPU-optimized tensor library developed for Deep Learning applications. It is a Python-based open-source machine learning program built mostly by Facebook's AI Research team. It is one of the most popular machine learning libraries, alongside TensorFlow and Keras. According to Google Search Trends, the PyTorch library is more popular than TensorFlow and Keras.

OpenCV is a free software library for computer vision and machine learning. OpenCV was developed to provide a common computer vision application infrastructure and rapidly incorporate machine perception into commercial products.

4.1.3 Hardware

We used Google colab pro with high-performance GPU of K80, P100 and T4. CPU 2xVCPU and RAM of 24GB. Colab worked faster with less processing time than other systems.

4.1.4 Evaluation Metric

Average Precision(AP) and Mean Average Precision(mAP) are the most popularly used metrics to evaluate object detection model like faster-RCNN, and Mask-RCNN. The mAP can be derived using AP[60].

Object detection evaluation comes down to accessing the correctness of detection.

True Positive(TP) is the model's accurate detection (TP).

False Positives (FP) are when a detector makes a mistaken identification.

False Negative (FN) is an object detector that misses a ground truth.

True Negative (TN) is The model that accurately failed to detect this background region. This measure is not utilized in object detection because such regions are not specifically marked while creating the annotations.

Intersection over Union:

Another helper metric called Intersection over Union(IOU) is required to define those concepts. When detecting objects, the IOU metric measures how well predictions(pd) and ground truth(gt) overlap. Any form, including a circle, a rectangular box, or an irregular shape, can be used for ground truth and prediction [60].

Figure 19 Intersection Over Union [58]

IOU evaluates ground truth(gt) and prediction(pd) overlap on a scale of 0 to 1, with 1 being perfect overlap. To decide if detection is accurate, we need a threshold, for example you can use this threshold along with IOU. A true positive (TP) in the context of an IOU (gt, pd) $\geq \alpha$, whereas a False Positive is evidence that IOU(gt, pd) $< \alpha$. A situation where IOU and ground truth values are not being considered, the results obtained from the system could be known as False Negatives as it disregarded (gt, pd) as the main parameters of the calculations.

Precision and Recall:

The degree to which the model correctly identifies just relevant things is known as precision (P). It shows the percentage of TPs among all the detections the model has generated. Equation (1) shows how precision is calculated.

$$P = \frac{TP}{TP + FP} \qquad (1)$$

Recall (R) is a metric used to assess a model's ability to recognize all ground truths or the notion of TPs among all ground truths shown using equation 2.

$$R = \frac{TP}{TP + FN} \qquad (2)$$

A model is deemed to be excellent when it has high precision and recall. A perfect model has precision = 1 and recall = 1, or zero FNs and zero FPs. A perfect model is usually impossible to achieve.

Precision x Recall Curve (PR Curve):

Plotting recall and precision at varying levels of confidence results in the precision-recall (PR) curve. The precision and recall of a good model remain high even with the confidence level changes.

Average Precision:

It is the Area under the precision and Recall Curve. Average Precision (AP) is being calculated at the IOU threshold α. High precision and recall yield a higher area under the curve. A unique AP score is determined for every class.

$$\mathrm{AP} = \int_0^1 p(r)\, dr \qquad (3)$$

<u>Mean Average Precision:</u>

The AP is calculated individually for every class, as discussed before. This indicates that there are the same numbers of AP values as there are classes. The mean of the average Precision (AP) values across all classes are known as the mean Average Precision (mAP).

$$\mathrm{mAP} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{AP}_i \qquad (4)$$

Using all these metric evaluations, the performance of object detection algorithms can be evaluated.

4.2 Image Annotation

Annotation of images is an essential component of computer vision; the technology obtains high-level understanding from images, videos, and audio to create a system that can behave and interpret like a human.

AI applications enabled by computer vision technology include autonomous cars, tumor detection, and unmanned aircraft. Nevertheless, without image annotation, the majority of these amazing computer vision applications would be infeasible. Annotation is another term for image tagging. Datasets must be integral components of computer vision machine learning and deep learning.

<u>PKLot:</u> The dataset is already annotated with the labelling of parking lots according to occupied and unoccupied spaces. Also, XML files in PASCAL VOC format were being downloaded.

<u>Custom Dataset:</u> The dataset is annotated using the LabelImg application. It is a tool for annotating images graphically. Annotations are recorded as XML files in PASCAL VOC format, which is the standard used by ImageNet (LabelImg label object bounding boxes in images). The graphic below illustrates a labelling example. We will trace through the parking lot and classify it as occupied or unoccupied at each intersection.

Figure 20 Image Annotation of Custom dataset

4.3 Data Pre-Processing

Data preparation or preprocessing, which comprises data filtering, enhancing, scaling, and feature extraction, is essential for developing an effective machine learning model. As described in Section 2.8.1, the preprocessing includes eliminating noise, pixel disturbances, and any other image-related flaws via various approaches.

With a few changes, the accuracy and mAP of this parking system were enhanced. Image Contrast Enhancement methods, such as dynamic histogram equalization, are implemented on both datasets.

In addition, data augmentation is performed prior to training our datasets using Mask-RCNN. Augmentations like Flipping, Gaussian Blurring and Affine were applied. The parameters within these augmentations can be changed accordingly. Figure 22 shows the applied augmentations on the image datasets, and the parameters are changed accordingly to try and find optimal results.

```
augmentation = imgaug.augmenters.Sometimes(0.1, [
                imgaug.augmenters.Fliplr(0.5),
                imgaug.augmenters.GaussianBlur(sigma=(0.0, 0.5)),
                imgaug.augmenters.Affine(
                    scale={"x": (0.8, 1.2), "y": (0.8, 1.2)},
                    translate_percent={"x": (-0.2, 0.2), "y": (-0.2, 0.2)},
                    rotate=(-25, 25),
                    shear=(-8, 8)
                )
            ])
```

Figure 21 Image Augmentation

Another approach implemented during Data preprocessing is image contrast enhancement. We implemented contrast enhancement techniques on PKLot dataset as it is comparatively a bigger dataset. PKLot is chosen as the chances for over-fitting is lesser with this dataset as it has a large number of images present.

Four separate datasets are being created from four different image contrast enhancement techniques explained before in subsection 2.7.3. We implemented Histogram equalization, Dynamic Histogram Equalization, Fusion exposure Framework and Contrast Limited AHE.

Examples of contrast-enhanced images:

Original Vs Histogram Equalization:

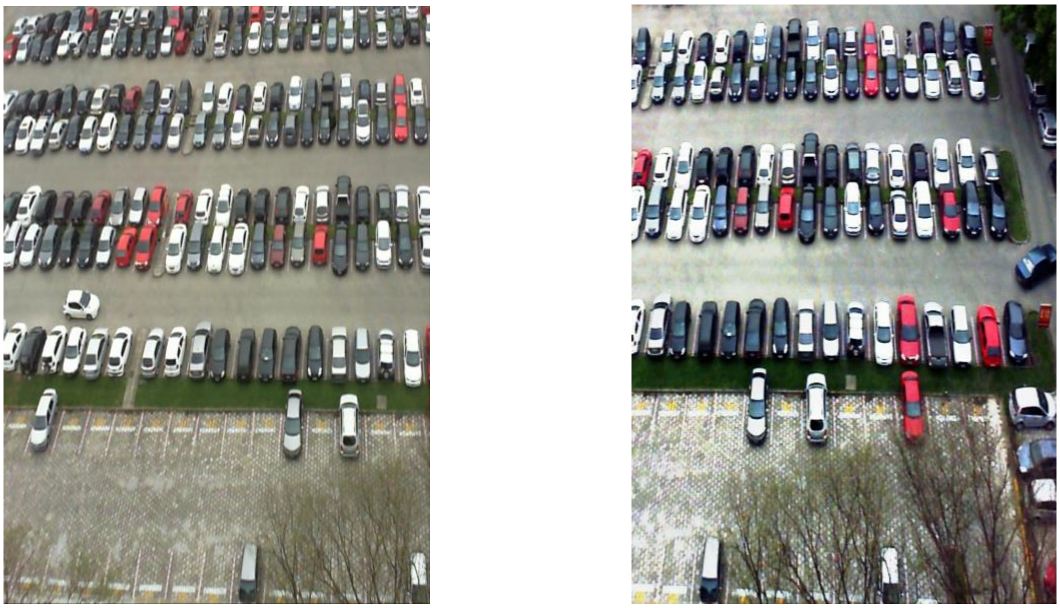


Figure 22 Original [59] vs Histogram Equalization Image

Original Vs Dynamic Histogram Equalization:

Figure 23 Original [59] vs Dynamic Histogram Equalization Images

Original Vs Contrast Limited AHE:

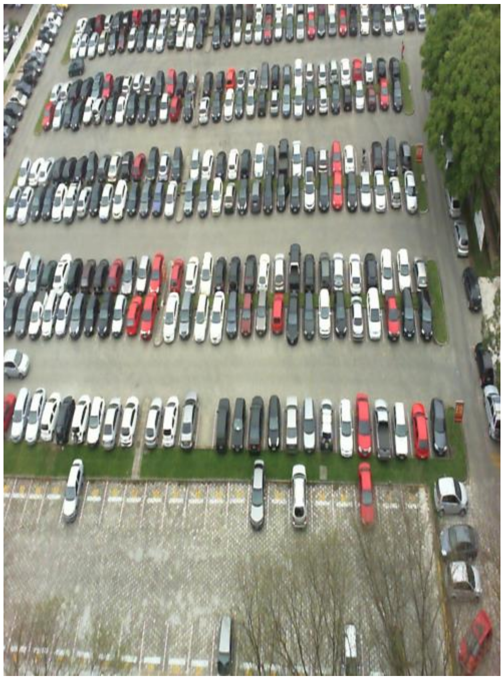
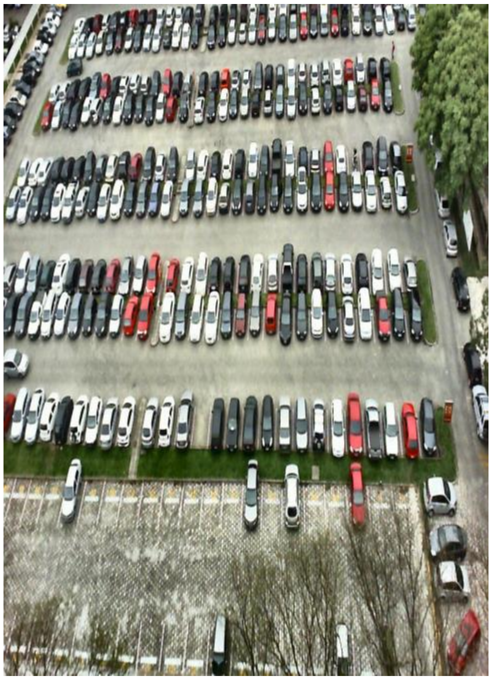
 

Figure 24 Original [59] vs Contrast Limited AHE

Original Vs Exposure Fusion Framework:

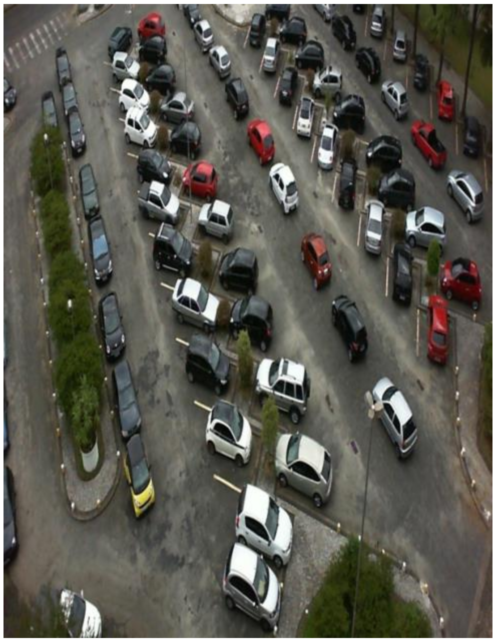
 

Figure 25 Original [59] vs Exposure Fusion Framework

4.4 Parking Detection System

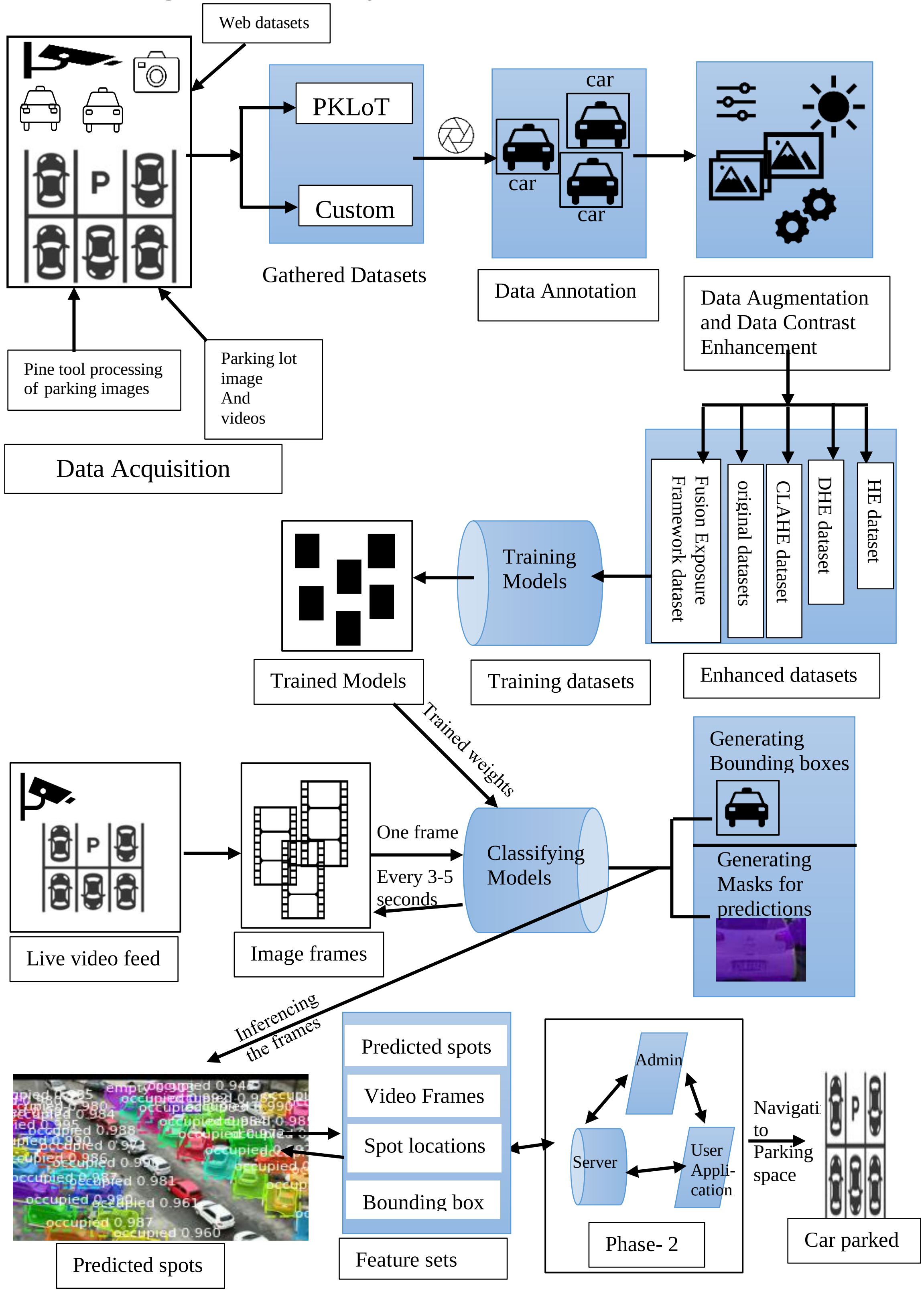


Figure 26 Parking Detection Architecture

Our proposed parking lot detection consists of two modules, as discussed below:

Using Faster-RCNN and Mask-RCNN, Phase 1 entails parking space recognition. The detection of parking is entirely space-driven. The collection comprises XML files containing annotated photos with occupied and unoccupied classes. Once the annotation is complete, preprocessing is performed, and the processed images are then transmitted for training to Faster-RCNN and Mask-RCNN. The training phase reveals how accurately the machine can forecast parking spaces. After training, the classification module is activated, the validation dataset is delivered into the system within 3-5 seconds, and classifiers work simultaneously on these images to generate predictions of parking space occupancy.

Figure 27 Flowchart of proposed Parking lot detection technique

Phase 2 commences as soon as the predictions are generated and entails the development of an algorithm for an android application that will be used to create an end-to-end smart parking system.

This section discusses phase 1 of the project. Phase 1 has two following modules, which we use as the base algorithm of our prediction process. Figure 21 depicts the overall framework of Parking lot detection on the real-time video feed, for dynamic and accurate results.

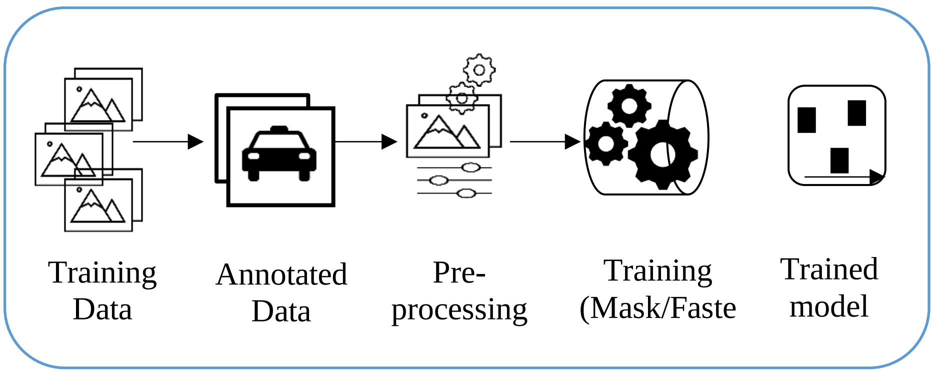


Figure 28 Module 1 Training Model

The framework of module 1 typically defines the Training phase of the system, it uses Mask-RCNN and Faster-RCNN for its training algorithm. It trains the system to detect the target class and differentiate it from the background. Also, to predict several instances within one image.

Figure 22 depicts the real-time parking lot detection module and the entire system architecture for the inference foundation.

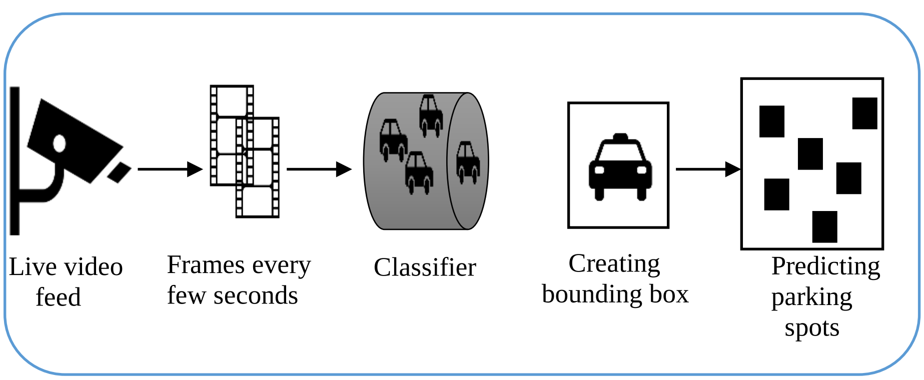


Figure 29 Module 2 Real Time parking detection

The model describes how it gets video from the feed and divides it into frames every few seconds to create a more realistic detecting system. The collected frames are then transmitted to the classifier for testing, after which the classifier creates bounding boxes over the predictions. As a result, we receive a picture or video including parking space predictions with labels indicating whether they are vacant or occupied.

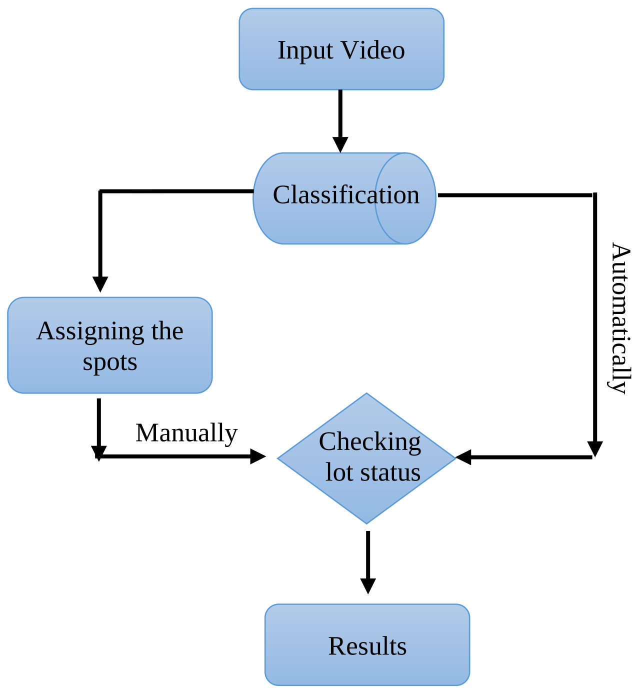


Figure 30 Algorithm Flowchart

Separate algorithms are employed to train the model:

- Faster-RCNN
- Mask-RCNN.

Faster-RCNN utilized the torch vision module of PyTorch, while Mask-RCNN utilized TensorFlow and Keras to generate optimal findings and predictions. Systems adopted a Space-based strategy: The system forecasts the availability of the parking lot. Since the initial model is trained to detect whether the spot contains a car or is available to the client, it can determine whether the spot is occupied or vacant. The predictions consist of a label, several spaces, and a bounding box around each place, with the corresponding classification.

4.5 End User Application

Phase 2 consists of a complete parking detection system architecture. We have developed a framework and initial blueprint for how the user will relate to the autonomous parking system.

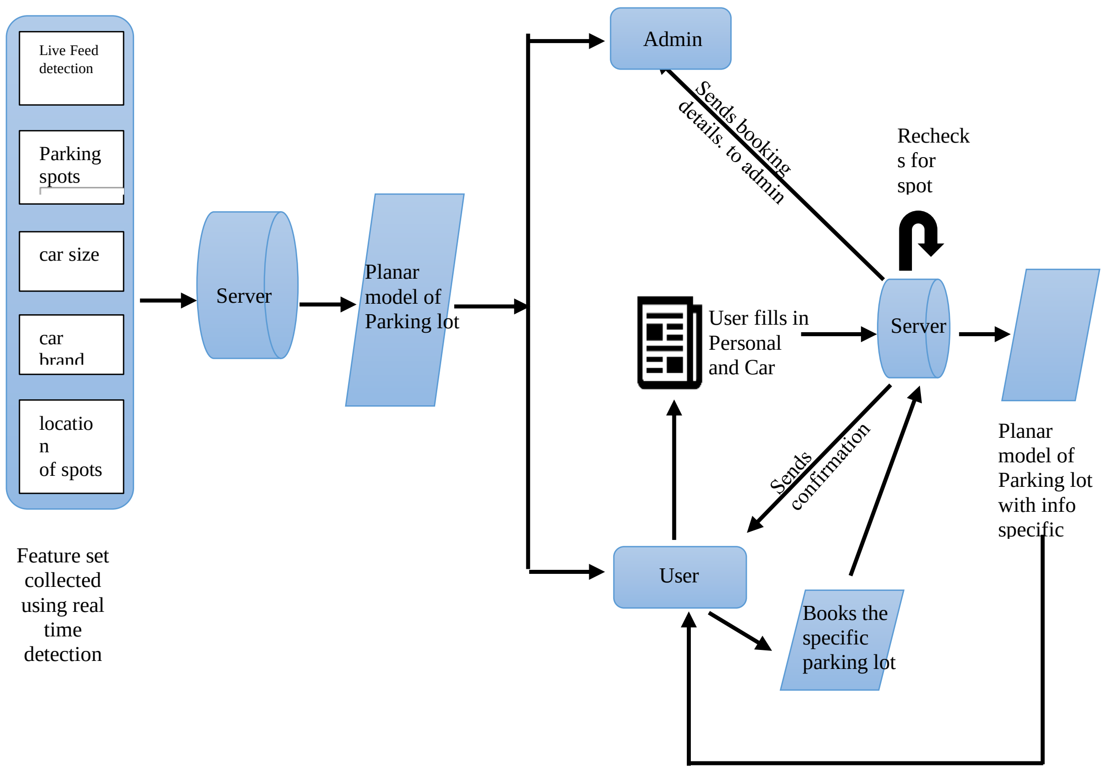


Figure 31 Architecture for Parking lot detection using a mobile application

The entire procedure shows how mobile application prototype architecture would process when all the building blocks are connected together. The objective of the entire procedure is to provide the user with a fully interactive, three-dimensional experience. The Android application will link the user to our servers. Within three modules, the preceding procedure is carried out.

Currently, the parking detection system has several building blocks that need to be integrated together to create a complete end-to-end application. The implemented parking lot detection method provides spot occupancy information only. As well as has a separate module of prototypes for the functionalities of the mobile application. The following system modules and its features will be created while extending this research.

The system will comprise of three modules:

- User Module: The primary user will download the application and provide both personal and vehicle information. The application will display various neighboring parking lots that employ our frameworks to the user. The user must select one parking lot before the program displays a complete animatic and planar image of the parking lot and a comprehensive mapping of the occupied and vacant parking spaces. In addition, the system will provide basic information about occupied parking spaces, such as the size , color and brand of the vehicle, depending on its detections and information. All this information will be dynamic and in sync with parking detection in real time.

  The user may wish to select a lot based on size or other criteria; these parameters will be provided to the server. The server will next examine the depth and highlight the parking spaces containing all these items.

  If a user does not want to select a slot themselves or cannot book any spot, the server will automatically select the spot and send the information further. If a user selects the location, and the spot is still available, the sever will book the lot. Once the booking is made, the user will receive the confirmation and the navigation towards their booked spot. If the time is about to run out for parking, the user would be asked 15 minutes prior, whether they want to extend their parking or leave.

- Server Module: This will serve as the engine of the entire system. The server will be responsible for retrieving data from the real-time detection system and generating an occupancy-marked planar view. Additionally, the server will be responsible for maintaining parking lot updates received from the detecting system and updating the system further.

  It will respond with the most recent results to queries from the user module. It will also verify that the reservation spot is still available; if it is not, it alerts the user to select another reservation spot. If the slot is available, the server will reserve it and update the feature set to prevent another user from claiming it.

The server module looks over the proper labelling of the parking spot and the number received from the classifier to its exact location with the GPS. This provides the exact navigation toward every spot. Also, it looks after following through with the transactions and fast deployments and results.

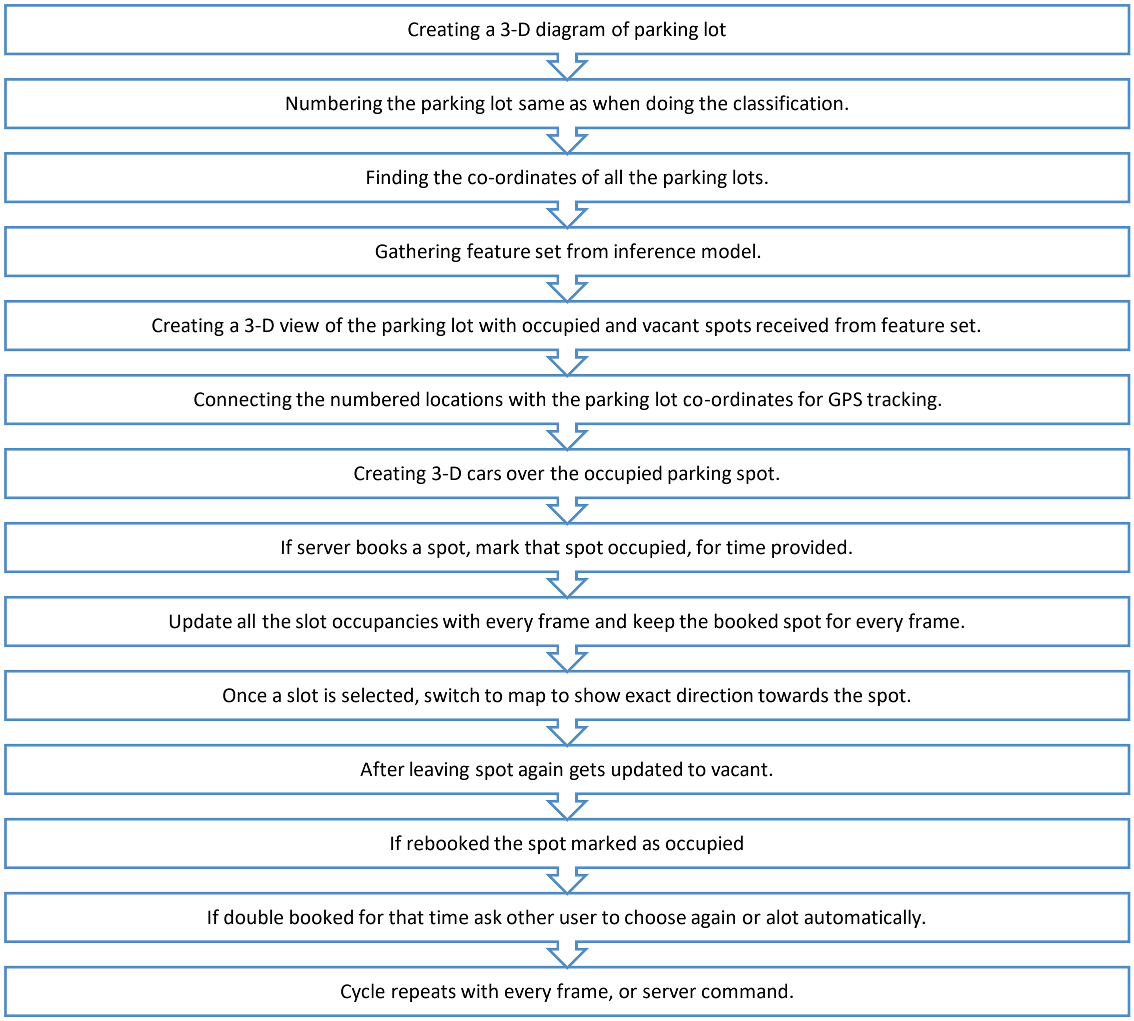
- Admin Module: This module will maintain a database of all logs and records. Also, will be used to verify the vehicle's information through its logs. As the cars leaving and entering are updated on the system, the admin will make sure it is well maintained. It provides further guidance to the user.

- Feature set: Developed system currently gives the status of parking lot occupancy. Whereas, Future work feature set would have the collection of features that would be extracted from the detection system. The feature set consists of parking lot regions, IOU, detected vehicles and the number of parking spaces available and occupied. It is the output we receive after the real-time detection is complete. Once the classifiers have worked on the input frames, it generates a set of results, including the location of the parking spot (numbered 1,2,3. in initial cases) and occupancy. If the spot is occupied, the system will also detect the car's color and size. A single frame contains all the details about every parking spot that is present there. These feature set changes with every passing frame for real-time detection and results.

- Planar Model: It will be a dynamic or planar view of the parking lot. This would show exactly what the parking lot looks like with labelled lots in a 3D view for easy understanding.

All these modules will create an end-to-end user-oriented parking lot detection system when put together.

4.5.1 Algorithm for Creation of Planar View of Parking lot



The above chart demonstrates the architecture for generating a three-dimensional picture of a parking lot within an application for the user's convenience. However, the algorithm is developed taking into consideration the functionalities that needs to be developed in the course of time. Within this algorithm, the parking detection system will be providing a large feature set to the server. However, the developed system is providing the occupancy status to the user. The feature set will include parking regions, vacant and occupied parking slots, and vehicle information for each occupied parking spot. This approach will be updated every three to five seconds to get a more precise picture of a parking lot. The server will build a three-dimensional model of the parking lot, including the number of empty and occupied spaces. To create a more realistic display, the program will also attempt to imitate life-like cars in the parking lot. Once these details have been completed, a model will be generated, and the server will maintain the information on reservations, cancellations, and completed current bookings. As soon as the reservation would be completed, the server would be responsible for marking the parking spot as occupied for the required period; if the reservation is cancelled, the parking space will return to the available slots automatically. Also, when the booking's time restriction is reached, the system will prompt the user to rebook or end their session. One of the functionalities for future works would also be that the application will inquire about any changes at least 20 minutes before the scheduled time. If the user wishes to rebook, the server will designate the slot as occupied again, and if it was already assigned, it would ask another user to pick a new space or let the server allocate it automatically. After these procedures are completed, the cycle will begin again with a new frame.

Chapter 5

5 Results and Observations

5.1 Faster-RCNN

This model works on torch- vision module of a PyTorch. Before working on the dataset, both datasets were pre-processed using the Image Enhancement technique called Dynamic Histogram equalization. We built a Faster-RCNN model using a pre-trained ResNet-50-FPN backbone. The model expects the input is intended to be a list of tensors of shape [C, H, W]. The model's behavior varies depending on whether it is in training or assessment mode. During training, the model demands both the input tensors and a set of targets (dictionary list), which includes labels and boxes. Whereas, during inference, the model only uses the input tensors and outputs the post-processed predictions as a dictionary list for each input image. The fields of the dictionary list are as follows, where N is the number of detections: boxes, labels, and scores. The model can be tweaked for different results using parameters.

Several parameters were modified during the experiment: max iter (iterations), warmup iters, base LR (Learning Rate), images per batch, gamma (learning rate decay), and batch size per image. The total number of training iters (iterations) is referred as the max iter, whereas warmup iters are untimed iterations that occur at the beginning. When attempting to minimize a loss function, the step size at each iteration is determined by base LR, which stands for base learning rate. If this step is too small, the learning rate may lead to a local minimum; if it is too large, the learning rate may bypass the function's ideal minimum. The batch size is defined as the number of training images used in a single iteration (ims per batch). Several testing with the following hyperparameter combinations yielded the best

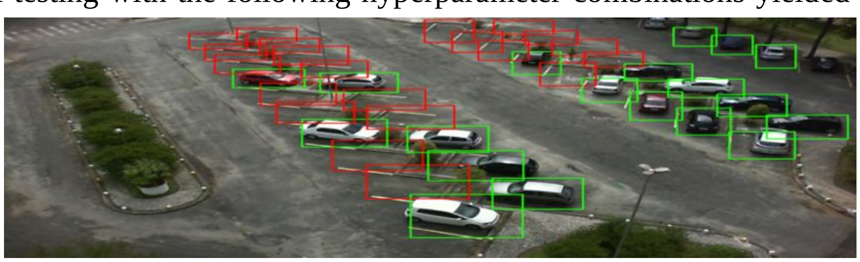


Figure 32 PKLot detection using Faster-RCNN

results: iterations = 1500, base learning rate = 0.001, images per batch = 1, gamma = 0.05. Analysis can be aided by examining training accuracy, training loss, validation accuracy and loss, model accuracy, and average precision (AP). Therefore, the total loss function

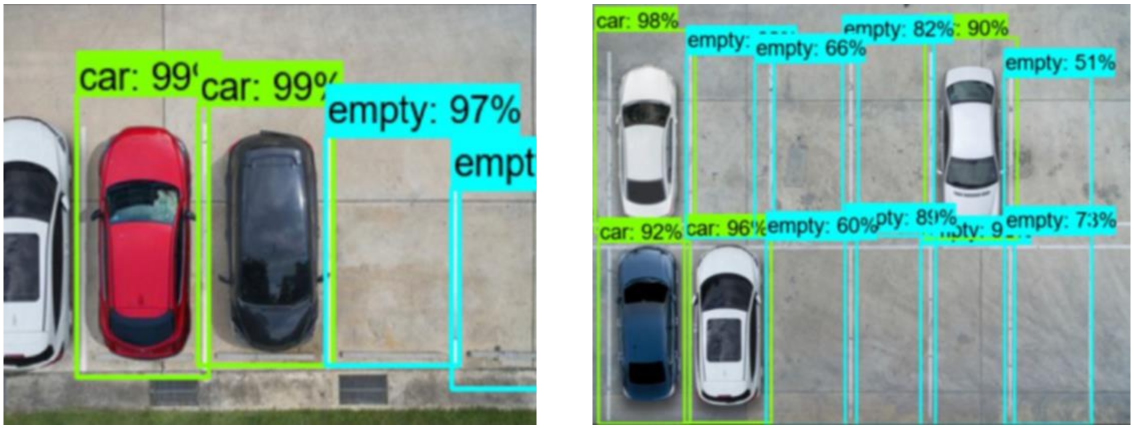


Figure 33 Custom Dataset detection using Faster-RCNN

shows no symptoms of overfitting and its value decreases over time, as predicted. Overall, the model's results are moderate.

The results indicate that the parking spot detections are somewhat accurate, although not for all. After training for approximately 3.5 - 4 hours, the results show an accuracy of 97.4%. After examining the PKLot dataset, the technique was implemented on our custom dataset, yielding an accuracy of 95.7%. Inferencing time for both of the datasets was 3 – 5 seconds. The prediction performance is affected by conditions such as image contrast or partially visible parking slots. It accurately detects automobiles and occupancy; it detects 7 out of 10 spot availability correctly. The epochs appear to be directly proportional to Average Precision (AP).

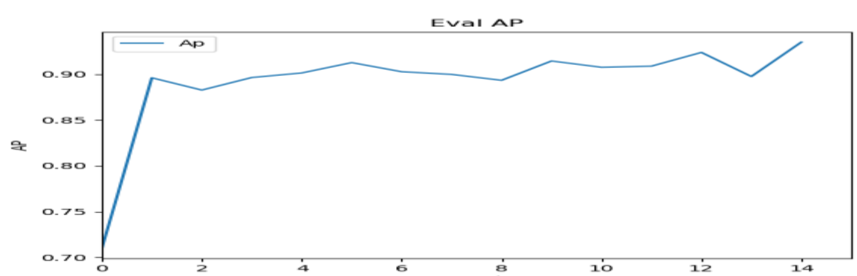


Figure 34 Relationship between AP and epochs

5.2 Mask-RCNN

This model is compatible with modules such as TensorFlow and Keras. The model is compatible with the ResNet101 backbone and ResNet50. We utilized the identical preprocessing method for the Mask-RCNN model. Additionally, we used augmentation before training to enlarge and diversify the dataset. We developed a space-based strategy for these algorithms, generating parking lot masks and instances based on their availability. The model is pre-trained using the COCO model. The fact that COCO dataset which contains eighty classes. The model's behavior differs depending on whether it is being trained or whether we are determining the inference.

We have used two different ResNet backbones to find the optimal architecture for our system. ResNet50 and ResNet101 are implemented on both datasets. The results were obtained for them with constraints of augmentation and no augmentations on datasets. Both frameworks were trained using Mask-RCNN with the following configurations GPU count =1, Images per GPU. = 1, whereas we are defining both occupied and vacant classes, therefore the overall number of classes is 2+1 (Background). The model is trained with two distinct learning rates of 0.001 with layers = “head” and 0.0001 with layers = ”all”, respectively. As we must achieve greater precision and decrease the loss function, we must prevent it from being trapped between local minima. Figure 31 below shows the random images from the training dataset with corresponding masks of occupied and vacant parking spots.

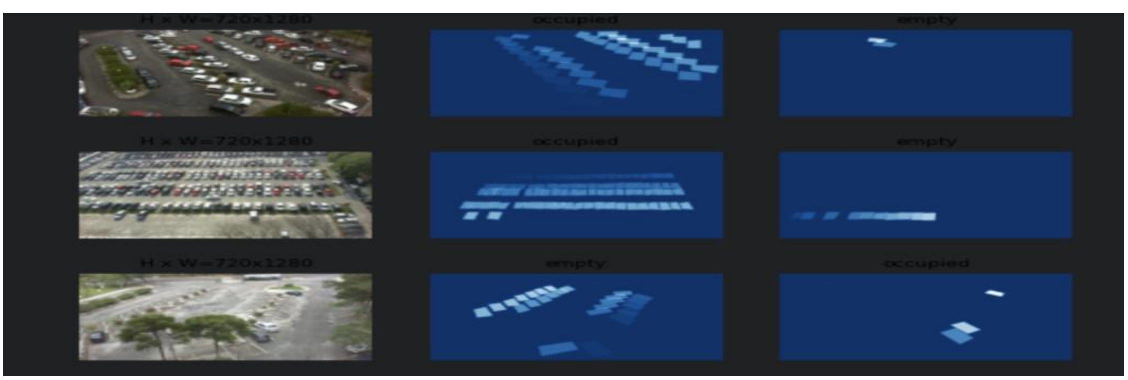


Figure 35 Creation of masks for Occupied and Empty Parking Spaces

Once the data is read, the model creates ground truth values for the masks for each class. Once the ground truth values have been sent and the masks have been created accordingly, the model is trained on these values to prepare it for inference. Training for the PKLot dataset takes 1.5 hours for epochs over 100 and approximately 10-30 minutes with epochs ranging from 30-100. The custom dataset takes around 25 minutes with epochs of around 100, due to the small size of the dataset. During model training, we observed a pattern in which the loss level decreased as the number of epochs increased. Figure 28 illustrates the inversely proportional (as the number of epochs increases, the loss function starts decreasing.) relationship between loss and epochs.

```
Epoch 1/60
100/100 [==============================] - 127s 1s/step - loss: 1.7613 - rpn_class_loss: 0.1698 -
Epoch 2/60
100/100 [==============================] - 61s 608ms/step - loss: 0.8350 - rpn_class_loss: 0.0360
Epoch 3/60
100/100 [==============================] - 62s 621ms/step - loss: 0.6558 - rpn_class_loss: 0.0273
Epoch 4/60
100/100 [==============================] - 62s 624ms/step - loss: 0.5645 - rpn_class_loss: 0.0222
Epoch 5/60
100/100 [==============================] - 73s 725ms/step - loss: 0.4853 - rpn_class_loss: 0.0183
Epoch 6/60
100/100 [==============================] - 76s 758ms/step - loss: 0.4756 - rpn_class_loss: 0.0165
Epoch 7/60
100/100 [==============================] - 77s 767ms/step - loss: 0.4243 - rpn_class_loss: 0.0148
Epoch 8/60
100/100 [==============================] - 77s 772ms/step - loss: 0.3821 - rpn_class_loss: 0.0135
Epoch 9/60
100/100 [==============================] - 76s 757ms/step - loss: 0.3522 - rpn_class_loss: 0.0112
Epoch 10/60
100/100 [==============================] - 78s 782ms/step - loss: 0.3262 - rpn_class_loss: 0.0110
Epoch 11/60
100/100 [==============================] - 76s 761ms/step - loss: 0.3221 - rpn_class_loss: 0.0091
Epoch 12/60
100/100 [==============================] - 78s 777ms/step - loss: 0.3193 - rpn_class_loss: 0.0081
```

Figure 35 Relationship between Epochs and loss

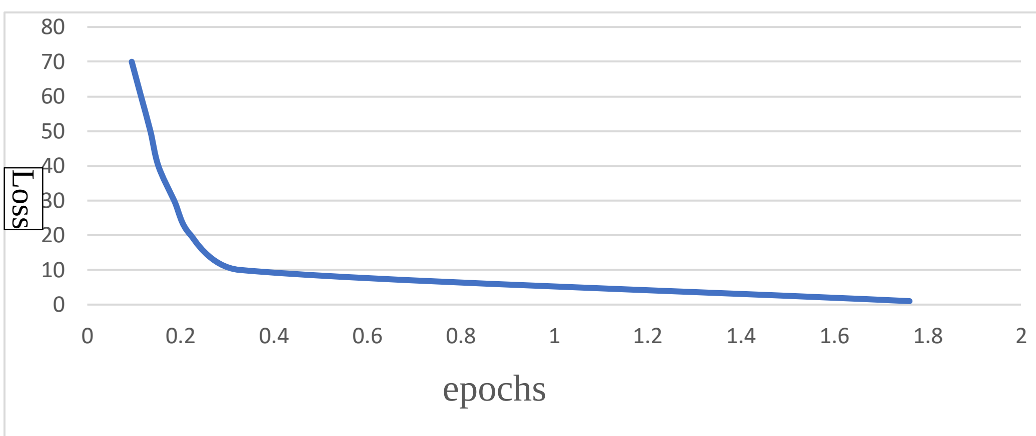


Figure 36 Relationship between Loss and Epochs

After completing training, the following observations regarding epoch and loss may be made. As the number of epochs increases, the loss reduces, making it inversely proportional to the epochs.

Table 6 ResNet50 Vs ResNet101 on PKLot

epochs	Steps per epoch	ResNet50	ResNet101	Augmentation	mAP
15+40	100	Yes		No	0.86541
30+40	100	Yes		No	0.82628
50+100	100	Yes		Yes	0.96801
50+100	200	Yes		Yes	0.91885
15+30	900	Yes		Yes	0.93071
25+40	900	Yes		Yes	0.9754
30+45	900	Yes		Yes	0.99656
30+45	100		Yes	No	0.95685
25+40	100		Yes	Yes	0.93052
40+61	100		Yes	Yes	0.9960
45+51	200		Yes	Yes	0.9990
3+6	900		Yes	Yes	0.9894
15+31	200		Yes	Yes	0.9552
11+20	500		Yes	No	0.9439

Table 7 ResNet50 Vs ResNet10 on custom dataset

epochs	Steps per epoch	ResNet50	ResNet101	Augmentation	mAP
15+40	100	Yes		No	0.683
30+40	100	Yes		Yes	0.749
50+100	900	Yes		Yes	0.923
80+100	500	Yes		Yes	0.932
50+100	500	Yes		Yes	0.804

15+40	100		Yes	No	0.730
25+40	900		Yes	Yes	0.975
30+45	900		Yes	Yes	0.927
80+100	100		Yes	Yes	0.893

Tables 6 and 7 show how different models performed with different architecture. It is observed that ResNet101 has outperformed ResNet50, and so for further implementation and improvements in our parking system, we use ResNet101 as our backbone.

Figure 36 Mask-RCNN predictions for ResNet101 on PKLot with mAP:0.9990

<u>Fine-tuning and Image Enhancement:</u>

Tables 5 and 6 show the results generated while using datasets and with minimum augmentations and basic configurations. We further implemented four types of Image processing algorithms on the dataset and created four different datasets. As observed in tables 5 and 6 ResNet101 gives more accurate results and PKLot can be extended further for fine-tuning the results. PKLot is a large dataset compared to a custom dataset, which helps decrease the false positive ratio. Also, PKLot performed better than the custom dataset during inference.

<u>Contrast enhancement Techniques:</u>

Histogram Equalization, DHE, CLAHE and exposure fusion framework techniques are applied to the dataset. Processed image datasets are saved separately in the drive. The system is further being trained on enhanced images using Mask-RCNN.

Configuration selected:

GPU_COUNT= 1

IMAGES_PER_GPU=1

NUM_CLASSES=1+2

STEPS_PER_EPOCH=100

DETECTION_MIN_CONFIDENCE=0.9

MAX_GT_INSTANCES=100

LEARNIN_RATE=0.001

IMAGE_MIN_DIM=128

IMAGE_MAX_DIM=1024

TRAIN_ROIS_PER_IMAGE=200

RPN_ANCHOR_SCALE=(32,64,128,256,512)

VALIDATION_STEPS=30

RPN_ANCHOR_RATIOS= [0.5,1,2]

Testing and Validation:

We used publicly available parking lot video and video created using custom validation data for testing. The video is further divided into frames and all the frames are stored in a folder. These frames are one by one sent into the inference with the trained weights generated by the model while training. The model tried to precisely predict the available parking slots and create a bounding box around them. The testing results are done on the validation dataset and images observed from frames together to check if the model works on all kinds of images, including noises and disturbances. The average inferencing time for both datasets is between 3 – 5 seconds.

Therefore, we observed the following results from our dataset.

Table 8 Image enhancement technique dataset Results

EPOCHS (LAYERS=" HEAD", L.R=0.001) + (LAYERS=" ALL", L.R=0.0001)	**HISTOGRAM EQUALIZATION(HE)**	**DYNAMIC HE (DHE)**	**EXPOSURE FUSION FRAMEWORK**	**Contrast Limited AHE (CLAHE) adaptive histogram equalization**
15+40	0.985836	0.998718	0.997500	0.900285
20+50	0.977938	0.997499	1.0	0.706071
30+50	0.999602	0.998718	0.995000	0.750438
45+70	0.995000	0.992500	1.0	0.859302
50+70	0.987900	0.995390	0.987437	0.953857
60+70	0.9	1.0	0.925879	0.904999

Table 8 shows the results obtained from 4 different datasets created from PKLot. Different epochs and, while training two different learning rates are being used for having deep layer training. In the first half of training, we try to train the top layers or heads of the model, and in the second phase, we try to work on all the layers.

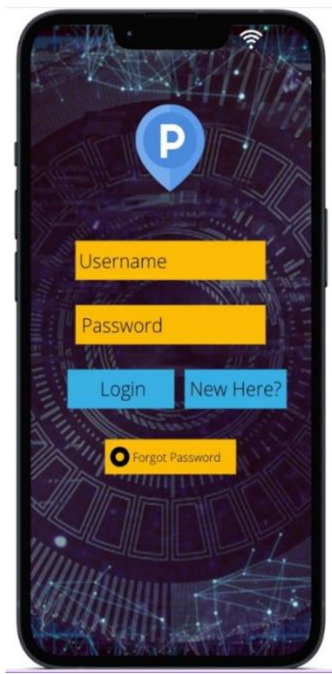
The results from enhanced images outperformed the results generated from PKLot with minor or no augmentations. Changing contrasts, exposure, or other factors have shown a significant change in training rate and accuracy. While PKLot gave a maximum mAP of 1.0, we can see dynamic histogram equalization and exposure fusion framework both gives mAP of 1.0 whereas prediction accuracy of ~ 98.85 for parking lot recognition using classification dataset.

Table 8 shows that the exposure fusion framework gives the best results giving mAP= 1.0 for validation sets. Our framework gives significant results compared to other similar works that give 0.9798 when they only had 66 images [56]. Furthermore, Sairam et al. [8] worked on PKLot gives 92.33 [8] prediction accuracy with 98.4% bbox prediction. In conclusion, our system provides higher accuracy with image enhancement techniques while utilizing less GPU resources and taking less training time.

5.3 Mobile Application Prototype

In this section, we demonstrate through the prototype of a mobile application that we developed. The application currently shows the dummy steps that will be followed by end-to-end mobile application. Different steps and functions of the prototype are shown within this section. The images/screens shown below explains the sequential functioning of the app for booking desired parking space. The prototype is the last building block of the detection system.

SCREEN-1



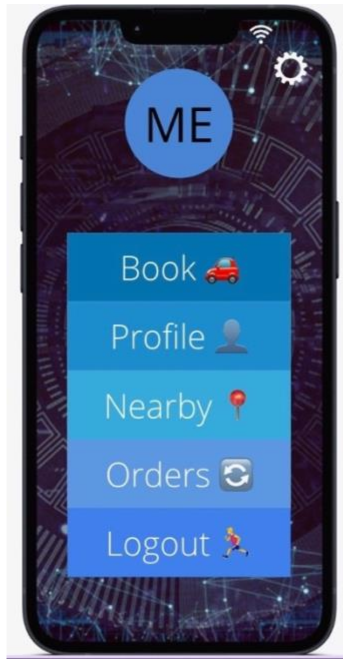
SCREEN 2

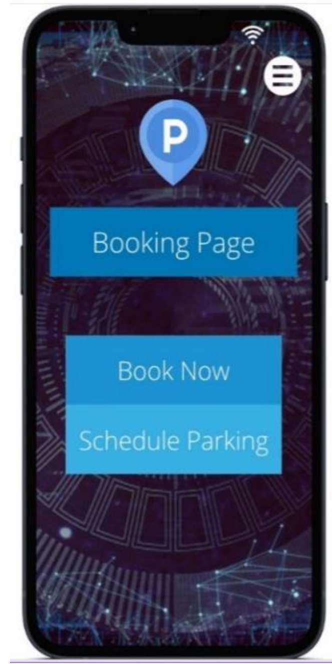
SCREEN 3

SCREEN-1: It shows how the app icon will look and it will be user-friendly and available across cross platforms.

SCREEN-2: It shows the login page for the application. It asks users if they already have an ID or if they want to create an ID. Users can log in and enter the application if they have an account.

SCREEN-3: It shows the information page for the user. First-time users will be signing into the app and filling up the form. Users will give their personal information and car details for a better experience.





SCREEN-4 SCREEN-5 SCREEN-6

SCREEN-4: The screen shows the profile created by the user with all its details.

SCREEN-5: It shows all the available list options within the application.

<u>Book:</u> Opens the booking manager for the user.

<u>Profile</u>: When clicked, shows the details about the user and vehicles.

<u>Nearby</u>: When clicked, gives all the nearby parking lots with occupancy and distance.

<u>Orders</u>: Shows the last booked parking lots and trips.

<u>Logout</u>: Helps users log out of the application.

SCREEN-6: Shows the booking page with booking options.

<u>Book Now</u>: Searches for parking spaces nearby that are available at the current time.

Schedule Parking: This helps the user to pre-book a parking spot at the desirable location at the needed time to save money, resources, and time.

SCREEN-7

SCREEN-8

SCREEN-9

SCREEN-7: It asks users about the time they want to park, which means the beginning of the parking meter and how many minutes and hours they would like to park at that spot. Once the selection is done the user can accept it by clicking on the DONE button and moving forward toward spot selection.

SCREEN-8: It shows nearby parking lots with vacancies for the given time. The user will have a list of lots sorted according to the increasing distance from his current location.

SCREEN-9: It shows the selected parking lot with its basic information like charges per lot, and the number of vacant spots for that time. And the distance to the parking lot. It also shows a map that when clicked google maps help with exact directions to the lot, with the expected time to reach the parking.

The button "BOOK" when clicked takes the user to another panel for booking and selecting the spot within the parking lot.

SCREEN-10 SCREEN-11 SCREEN-12

SCREEN-10: This screen will show the user a live 360degree and 3-D animatic view of the parking lot with all the parking spots and parked cars within the facility. This view will also show a 3-D view of cars, which can help users know what all types of cars and their brands are parked (like trucks, SUV and cars). This feature can easily help the users to visualize their surroundings and choose which parking spot is a better option for them. The view shows the available parking slots with slightly elevated areas for the space and with a sign "P".

SCREEN-11: This screen shows the selected slot by the user with an elevated green selection and shows the spot with the user's car animation over it. So, the user can visualize how their vehicle will look parked within the spot with surrounding cars.

SCREEN-12: This screen shows the complete booking details with the spot selected, charges, slot time and parking lot name. It also shows the user details for parking tickets. The screen consists of three buttons, Back, Edit and Pay Now. "Back" helps the user to go back and choose the parking spot again. Edit helps in editing the parking lot detection as it takes the user back to SCREEN-8. The Pay Now button directs the user to just pay for the ticket and move towards the spot.

SCREEN-13

SCREEN-13: This screen shows a QR Code that is generated after the payment, which will be scanned at the entrance of the parking lot. The QR code is like an access pass which is generated with all our parking details. This screen also shows the navigation details toward the exactly selected parking slot. It will be shown with the help of google maps. The let’s go button will open the maps application and direct the user toward the location.

Chapter 6

6 Discussions and Conclusion

6.1 Limitations and Assumptions

- In order to deploy the proposed parking detection solution in an actual parking lot, all the lot users need to use and book through the mobile application.

- The system is being trained on normal and mild weather conditions, it can underperform during severe conditions like heavy snowfalls, and hefty rainfalls.

- The technique is trained over detecting occupied or vacant space, which can falsely detect smaller objects and occluded spaces as occupied.

- Designed parking detection system accurately detects on public datasets with less training. However, the system needs more training data and training time for the custom datasets.

- During this research, limited algorithms (Faster-RCNN and Mask-RCNN) and backbones (ResNet50 and ResNet101) are being used

6.2 Conclusion and Future Works

Our proposed smart parking system is designed with the user's challenges in locating a parking spot in mind. The system intends to provide an effective, affordable, and accurate parking solution to help end users and lot operators. The primary objective is to develop a real-time parking spot detection and response system. In every frame, the system identifies occupied and vacant parking spaces. In the first experiment, the Faster-RCNN and Mask-RCNN algorithms were used. We utilized two separate parking lot datasets. The datasets were preprocessed with dynamic histogram equalization, followed by additional training. After training on both datasets, Faster-RCNN produces a detection accuracy of 97.4%. We then developed a Mask-RCNN system, which provides better accuracy than Faster-RCNN. Mask-RCNN yielded results as mAP of 0.9990 when worked with contrast-enhanced images. Mask-RCNN generates a mask for each detection based on its label and its bounding boxes. In contrast, it was evident that Mask-RCNN produces better and faster detection when datasets are trained on enhanced images. The labelling and detection of outcomes are better than Faster-RCNN. When compared with the preprocessed model of Naufal et al. [47] their IOU accuracy was 85.80 and using mAlexNet gave them an accuracy of 73.73%. Therefore, our systems show a significant increase in detection accuracy and mAP compared with their works.

We have developed several stepping stones for the parking detection system. In this thesis, we have covered the algorithm and architecture for collecting data from the system. The accomplished system is capable of detecting parking space occupancy accurately. Furthermore, We also develop a prototype of an Android application that explains the functionalities and working of the application. The prototype shows how the application allows the user to reserve a parking spot and travel accordingly. The system will get information and specifics about the occupancy, location, and neighbours, and then displays all neighbouring parking spaces that are suitable for cars of comparable dimensions. Once the reservation has been made, the QR code for entrance is generated, and the app will direct the user to the reserved parking spots. As part of future work, all of these blocks need to be integrated together to construct a complete system. So, once the whole system is developed and connections are established between all these blocks, our backend system would provide the parking spot detection; and provides the user with the parking spot availability information, our frontend i.e., android App will allow them to book and navigate to their parking spots.

For future studies, we plan to fine-tune the hyper-parameters to improve both methods' detection accuracy while reducing training time. Also, we plan to implement systematic and optimal methods for adjusting and fine-tuning the hyper-parameters. Furthermore, gathering more images for datasets, including severe weather conditions (such as snow, heavy rainfall, and storms) is in the future plan. This will allow us to train the models to detect parking occupancy during different weather conditions and with occlusions (small objects, cycles, person etc.). Also, designing a system that can detect vehicles' make, size, and the color is another future direction of this research. To further extend our research, we would also like to explore several other algorithms (such as YOLO, SSD, MobileNet etc.), backbones and image enhancement techniques for better comparative analysis. For cross-validating the predicted result, an algorithm to check if the results deduced are correct will also be worked upon. Last but not least, developing an end-to-end mobile application for parking spot detection is one of the main objectives of our future research.

References

1) Walch, K. (2020, May 31). The autonomous systems pattern of ai. Forbes. Retrieved January 22, 2022, from https://www.forbes.com/sites/cognitiveworld/2020/05/30/the-autonomous-systems-pattern-of-ai/

2) Stanford sites. Stanford Sites | University IT. (n.d.). Retrieved January 22, 2022, from https://uit.stanford.edu/service/stanfordsites

3) Haddal, R., & Hayden, N. K. (2018, September 1). Autonomous Systems Artificial Intelligence and safeguards. Autonomous Systems Artificial Intelligence and Safeguards. (Conference) | OSTI.GOV. Retrieved January 22, 2022, from https://www.osti.gov/servlets/purl/1561151

4) Hewlett Enterprise. (n.d.). What is Artificial Intelligence (AI)?: Glossary. What is Artificial Intelligence (AI)? | Glossary | HPE Canada. Retrieved January 22, 2022, from https://www.hpe.com/ca/en/what-is/artificial intelligence.html

5) Yoshua, B. (2009). Learning Deep Architectures for AI. Foundations and Trends in Machine Learning 2(1): 1-127.

6) K. McCoy, “Drivers spend an average of 17 hours a year searching for parking spots,” 2017.

7) J. Ahmad, Z. Lewis, P. Duraisamy and T. Mcdonald, "Parking Lot Monitoring using MRCNN," 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2019, pp. 1-4, doi: 10.1109/ICCCNT45670.2019.8944394.

8) B. Sairam, A. Agrawal, G. Krishna and S. P. Sahu, "Automated Vehicle Parking Slot Detection System Using Deep Learning," 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), 2020, pp. 750-755, doi: 10.1109/ICCMC48092.2020.ICCMC-000140.

9) Agrawal, T., & Urolagin, S. (2020, January 1). Multi-angle parking detection system using mask R-CNN: Proceedings of the 2020 2nd International Conference on Big Data Engineering and Technology. ACM Other conferences. Retrieved July 22, 2022, from https://dl.acm.org/doi/pdf/10.1145/3378904.3378914

10) Winston, P. H. (1992). Artificial intelligence. Addison-Wesley Longman Publishing Co., Inc.

11) Hastie, T., Tibshirani, R., & Friedman, J. (2009). Overview of supervised learning. In The elements of statistical learning (pp. 9-41). Springer, New York, NY

12) Khanum, M., Mahboob, T., Imtiaz, W., Ghafoor, H. A., & Sehar, R. (2015). A survey on unsupervised machine learning algorithms for automation, classification and maintenance. International Journal of Computer Applications, 119(13).

13) Dixit, M., Tiwari, A., Pathak, H., & Astya, R. (2018, October). An overview of deep learning architectures, libraries and its applications areas. In 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN) (pp. 293-297). IEEE.

14) Lisowski, E. (2022, March 14). Deep learning architecture. Addepto. Retrieved April 30, 2022, from https://addepto.com/deep-learning-architecture/

15) Karpathy, A. (2017). Convolutional neural networks for visual recognition. Notes accompany the Stanford CS class CS231.

16) Hubel, D. H. (1995). Eye, brain, and vision. Scientific American Library/Scientific American Books.

17) Ramprasath, M., Anand, M. V., & Hariharan, S. (2018). Image classification using convolutional neural networks. International Journal of Pure and Applied Mathematics, 119(17), 1307-1319.

18) Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

19) Ren, S., He, K., Girshick, R., & Sun, J. (2016, January 6). Faster R-CNN: Towards real-time object detection with region proposal networks. arXiv.org. Retrieved August 30, 2022, from https://arxiv.org/abs/1506.01497

20) He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2018, January 24). Mask R-CNN. arXiv.org. Retrieved August 30, 2022, from https://arxiv.org/abs/1703.06870

21) Szeliski, R. (2010). Computer vision: algorithms and applications. Springer Science & Business Media.

22) Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. J Big Data 6, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0

23) M. Abdullah-Al-Wadud, M. H. Kabir, M. A. Akber Dewan and O. Chae, "A Dynamic Histogram Equalization for Image Contrast Enhancement," in IEEE Transactions on Consumer Electronics, vol. 53, no. 2, pp. 593-600, May 2007, doi: 10.1109/TCE.2007.381734.

24) Equalization, A. H. (2014, September). accessed online from" http://en. wikipedia. org/wiki/Adaptive_histo gram_equalization. Date.

25) Ying, Z., Li, G., Ren, Y., Wang, R., & Wang, W. (2017, August). A new image contrast enhancement algorithm using exposure fusion framework. In International conference on computer analysis of images and patterns (pp. 36-46). Springer, Cham.

26) B. Y. Cai, R. Alvarez, M. Sit, F. Duarte and C. Ratti, "Deep Learning-Based Video System for Accurate and Real-Time Parking Measurement," in IEEE Internet of Things Journal, vol. 6, no. 5, pp. 7693-7701, Oct. 2019, doi: 10.1109/JIOT.2019.2902887.

27) Basavaraju S R. (2015) .Automatic Smart parking System using Internet of Things (IOT).International Journal of Scientific and Research Publications, Volume 5, Issue 12, December 2015

28) Gandhi,B.M.K.,&Rao,M.K.(2016).APrototypeforIoTbasedCar Parking Management system for Smart cities. Indian Journal of Science and Technology, 9(17).

29) Patil, M., & Bhonge, V. N. (2013). Wireless sensor network and RFID for smart parking system. International Journal of Emerging Technology and Advanced Engineering, 3(4), 188-192.

30) Singh, H., Anand, C., Kumar, V., & Sharma, A. Automated Parking System With Bluetooth Access. International Journal Of Engineering And Computer Science ISSN, 2319-7242.

31) Orrie, O., Silva, B., & Hancke, G. P. (2015, November). A wireless smart parking system. In Industrial Electronics Society, IECON 2015- 41st Annual Conference of the IEEE (pp. 004110-004114). IEEE.

32) D. Thomas, B. C. Kovoor “A Genetic Algorithm Approach to Autonomous Smart Vehicle Parking System” 6th International Conferences on Smart Computing and Communications, pp.68-76 2017.

33) Karbab, E., Djenouri, D., Boulkaboul, S., & Bagula, A. (2015, May). Car park management with networked wireless sensors and active RFID. In 2015 IEEE International Conference on Electro/Information Technology (EIT) (pp. 373-378). IEEE.

34) Al-Kharusi, H. and Al-Bahadly, I. (2014) Intelligent Parking Management System Based on Image Processing. World Journal of Engineering and Technology, 2, 55-67. doi: 10.4236/wjet.2014.22006.

35) Orhan Bulan, Robert P. Loce, Wencheng Wu, Yao Rong Wang, Edgar A. Bernal, Zhigang Fan, "Video-based real-time on-street parking occupancy detection

system," J. Electron. Imag. 22(4) 041109 (12 August 2013)https://doi.org/10.1117/1.JEI.22.4.041109

36) Fusek, R., Mozdřeň, K., Šurkala, M., & Sojka, E. (2013). Adaboost for parking lot occupation detection. In Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013 (pp. 681-690). Springer, Heidelberg.

37) Nurullayev, S., & Lee, S. W. (2019). Generalized parking occupancy analysis based on dilated convolutional neural network. Sensors, 19(2), 277.

38) Acharya, D., Yan, W., & Khoshelham, K. (2018). Real-time image-based parking occupancy detection using deep learning. Research@ Locate, 4, 33-40.

39) Mora, J. E. G., Lopera, J. C. L., & Cortes, D. A. P. (2018). Automatic visual classification of parking lot spaces: A comparison between bof and cnn approaches. In Workshop on Engineering Applications (pp. 160–170). Springer.

40) Dhuri, V., Khan, A., Kamtekar, Y., Patel, D., & Jaiswal, I. (2021). Real-time parking lot occupancy detection system with vgg16 deep neural network using decentralized processing for public, private parking facilities. In 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT) (pp. 1–8). doi:10.1109/ICAECT49130.2021.9392506.

41) Valipour, S., Siam, M., Stroulia, E., & Jagersand, M. (2016). Parking-stall vacancy indicator system, based on deep convolutional neural networks. In 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT) (pp. 655–660). IEEE.

42) G. Amato, F. Carrara, F. Falchi, C. Gennaro and C. Vairo, "Car parking occupancy detection using smart camera networks and Deep Learning," 2016 IEEE Symposium on Computers and Communication (ISCC), 2016, pp. 1212-1217, doi: 10.1109/ISCC.2016.7543901.

43) Nyambal, J., & Klein, R. (2017). Automated parking space detection using convolutional neural networks. 2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech), 1-6.

44) Rahman, S., Ramli, M., Arnia, F., Sembiring, A., & Muharar, R. (2020). Convolutional neural network customization for parking occupancy detection. In 2020 International Conference on Electrical Engineering and Informatics (ICELTICs) (pp. 1–6). IEEE.

45) Mart´ın Nieto, R., Garc´ıa-Mart´ın, A´., Hauptmann, A. G., & Mart´ınez, J. M. (2019). Automatic vacant parking places management system using multi- camera vehicle detection. IEEE Transactions on Intelligent Transportation Systems, 20, 1069–1080.

46) Poply, P., & Shetty, S.D. (2019). Urbanised Real-time Parking Detection System using Deep Learning. Proceedings of the 2019 3rd International Conference on Computer Science and Artificial Intelligence

47) Naufal, Ahmad Ridhwan, Chastine Fatichah and Nanik Suciati. “Preprocessed Mask RCNN for Parking Space Detection in Smart Parking Systems.” International Journal of Intelligent Engineering and Systems 13 (2020): 255-265.

48) Xu, C., Wang, G., Yan, S., Yu, J., Zhang, B., Dai, S., Li, Y., & Xu, L. (2020). Fast Vehicle and Pedestrian Detection Using Improved Mask R-CNN. Mathematical Problems in Engineering, 2020, 1-15.

49) Gabzdyl, D. (2020). Counting Vehicles in Image and Video.

50) Gregor, M., Pirník, R., & Nemec, D. (2019, July 30). Transfer learning for classification of parking spots using residual networks. Transportation Research Procedia. Retrieved August 31, 2022, from https://www.sciencedirect.com/science/article/pii/S2352146519303527

51) Mettupally, S. N. R., & Menon, V. (2019). A smart eco-system for parking detection using deep learning and big data analytics. In 2019 SoutheastCon (pp. 1–4). IEEE.

52) Agrawal, T., & Urolagin, S. (2020, January 1). Multi-angle parking detection system using mask R-CNN: Proceedings of the 2020 2nd International Conference on Big Data Engineering and Technology. ACM Other conferences. Retrieved July 22, 2022, from https://dl.acm.org/doi/pdf/10.1145/3378904.3378914

53) G. Khan, M. A. Farooq, Z. Tariq and M. U. G. Khan, "Deep-Learning Based Vehicle Count and Free Parking Slot Detection System," 2019 22nd International Multitopic Conference (INMIC), 2019, pp. 1-7, doi: 10.1109/INMIC48123.2019.9022687

54) Developing a Parking Monitoring System using Mask-RCNN Siddharth Majgaonkar1 *, Karan Waghela2 , Raj Shah3 , Aruna Pavate4, Priti Mishra5

55) Nguyen, T., Tran, T., Mai, T., Le, H., Le, C., Pham, D., & Phung, K.-H. (2021). An adaptive vision-based outdoor car parking lot monitoring system. In 2020 IEEE Eighth International Conference on Communications and Electronics (ICCE) (pp. 445–450). IEEE.

56) García, J. M. C., Piorno, J. R., & Mata-Garcia, M. G. (2020, July 1). Automatic detection of vehicles in outdoor parking lots from Zenith perspective using neural networks: Proceedings of the 2020 Summer Simulation Conference. ACM Other conferences. Retrieved July 22, 2022, from https://dl.acm.org/doi/10.5555/3427510.3427526

57) R. Vîlceanu, M. Oniţa and A. Ternauciuc, "Analyzing parking lots vacancy detection algorithms using Mask R-CNN implementations," 2020 International Symposium on Electronics and Telecommunications (ISETC), 2020, pp. 1-4, DOI: 10.1109/ISETC50328.2020.9301141.

58) Y. Song, J. Zeng, T. Wu, W. Ni and R. P. Liu, "Vision-Based Parking Space Detection: A Mask R-CNN Approach," 2021 IEEE/CIC International Conference on Communications in China (ICCC), 2021, pp. 300-305, DOI: 10.1109/ICCC52777.2021.9580236.

59) De Almeida, P. R.; Oliveira, L. S.; et al. PKLot–A robust dataset for parking lot classification. Expert Systems with Applications, volume 42, no. 11, 2015: pp. 4937–4949.

60) R. Padilla, S. L. Netto and E. A. B. da Silva, "A Survey on Performance Metrics for Object-Detection Algorithms," 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), 2020, pp. 237-242, doi: 10.1109/IWSSIP48289.2020.9145130.

61) Rama, K., Kumar, P., & Bhasker, B. (2019). Deep Learning to Address Candidate Generation and Cold Start Challenges in Recommender Systems: A Research Survey. ArXiv, abs/1907.08674.

62) De Almeida, P. R., Oliveira, L. S., Britto Jr, A. S., Silva Jr, E. J., & Koerich, A. L. (2015). PKLot–A robust dataset for parking lot classification. Expert Systems with Applications, 42(11), 4937-4949.

63) Y. Song, J. Zeng, T. Wu, W. Ni and R. P. Liu, "Vision-Based Parking Space Detection: A Mask R-CNN Approach," 2021 IEEE/CIC International Conference on Communications in China (ICCC), 2021, pp. 300-305, doi: 10.1109/ICCC52777.2021.9580236.

64) Pine tools. PineTools. (n.d.). Retrieved September 2, 2022, from https://pinetools.com/

65) Contact us Susan Merkell Director | Finance Transformation smerkell@deloitte.se . (2020, March 17). Part 1: Artificial Intelligence defined: Deloitte: Technology Services. Deloitte Sweden. Retrieved September 2, 2022, from

https://www2.deloitte.com/se/sv/pages/technology/articles/part1-artificial-intelligence-defined.html

66) Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).

67) Turing. (2022, May 2). Deep Learning vs Machine Learning: The ultimate battle. Deep Learning vs Machine Learning. Retrieved September 3, 2022, from https://www.turing.com/kb/ultimate-battle-between-deep-learning-and-machine-learning

Curriculum Vitae

Name:	Aakriti Sharma
Post-secondary	2014 – 2018: B.Tech in Information Technology
Education and	K.I.E.T Group Of Institutions, Ghaziabad, India
Degrees:	2020-2022: M.Sc. in Computer Science
	The University of Western Ontario
Honours and	
Awards:	
Related Work	Teaching/Research Assistant
Experience	The University of Western Ontario
	2020-2022

Publications:

S. Saha, A. T. Priyoti, A. Sharma and A. Haque, "Towards an Optimal Feature Selection Method for AI-Based DDoS Detection System," 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), 2022, pp. 425-428, DOI: 10.1109/CCNC49033.2022.9700569.