Volume 9, Issue 4, April – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR234

Parking Occupancy Detection using Computer Vision Techniques

Ashish Katiyar[1]
Geoinformatics, Department of Civil Engineering
Indian Institute of Technology, Kanpur
Uttar Pradesh, India

Seema[2]
M.Tech., Department of Civil Engineering
Madan Mohan Malaviya University of Technology
Gorakhpur Uttar Pradesh, India

Abstract:- The surge in vehicle numbers on roads contributes significantly to traffic congestion and management challenges, particularly evident in developing nations like India where the influx of cars exceeds road and parking capacity. Addressing these issues necessitates the implementation of sophisticated parking management systems. This project focuses on two key objectives: detecting vehicle occupancy within marked parking slots and analyzing parking data. Using the parking lot near IIT Kanpur main gate as a reference, video data was collected for 14 consecutive days, enabling the evaluation of vehicle occupancy and parking patterns.

Object detection algorithms such as Mask-RCNN and YOLO-v5 were employed to identify occupied parking spaces within the lot. Various methods, including HAAR cascade-based classifiers, DNN-based systems utilizing ResNet classifiers, and RCNN with IoU, were tested for detecting vehicles within allotted slots. The data collected was stored in CSV format for analysis.

This project aims to provide insights into detecting parking space availability and analyzing parking data to optimize time and fuel efficiency. In the Mask-RCNN approach, pre-occupied spaces are denoted by red boxes, while green boxes represent available parking spots. Similarly, YOLOv5 was utilized to count cars in video frames and identify available parking spaces. The YOLO Annotation Toolbox facilitated the extraction of parking space coordinates from recorded video frames, which were then visualized in QGIS for further analysis.

Keywords:- *Parking Management System, Object Detection Algorithms, Mask-RCNN, Yolov5, Data Analysis, Q GIS, Real-Time Data Handling, Computer Vision, Urban Mobility.*

I. INTRODUCTION

The automobile industry report highlights India's remarkable growth from producing 22.93M vehicles annually in 1733 to a US $32.73B car market by FY 2021, projected to reach US $54.8B by FY 2027 [1]. This growth is paralleled by an increase in vehicles, commercial buildings, and connecting roadways, making transportation hubs essential to modern society.

However, with the surge in vehicles, traffic congestion and parking problems have escalated. Modern transportation infrastructure and parking facilities struggle to keep pace, exacerbated by India's status as a developing country. Challenges include insufficient parking lots and a narrower road network compared to developed countries.

High-density cities like Delhi and Mumbai face acute parking shortages, leading to vehicles occupying roadsides, compounding congestion and safety risks. Finding available parking becomes time-consuming, causing traffic to overflow as drivers search for spots [2].

Presently, manual parking procedures persist, contributing to fuel and time wastage. Inefficient planning and management further exacerbate the issue, impacting public places' viability as people avoid them due to parking constraints.

A smart parking system, a component of India's Intelligence Transportation System (ITS), integrates various technologies such as communication, information processing, electronics, and control. Incorporating these technologies into India's transportation system yields numerous benefits. Advancements like radio frequency identification (RFID), Internet of Things (IoT)-based solutions, and artificial intelligence (AI) methods, including machine learning, deep learning, and computer vision, have been successfully implemented [3]. This research project focuses on designing a cost-effective automated off-street parking system to enhance efficiency for patrons.

The main objective of this work is to develop a system by which the parked and unoccupied spaces within the parking lot can be detected. It's also known that multiple automated parking systems are already designed and implemented but those systems cannot be applied in India directly. So, developing a cost-effective parking system is the main objective of this project, where the use of sensors is limited as the maintenance cost of those sensors is extremely high. Instead of this, there are the following sub-objectives of this work.

- Detection of the vehicles within the video frame using a Camera sensor.
- Locate the parking spaces in the marked as well as unmarked parking lots.
- Identify the presence or absence of the vehicles within the allotted parking space.
- To find patterns and insights by parking data analysis.

II. LITERATURE REVIEW

In this section, we review previous research on automatic smart parking management systems. The study aims to offer insights into the extent of previous work and the diverse approaches employed in designing and implementing such systems. Various methods have been utilized across different studies to address unique challenges such as parking space availability, traffic congestion, weather conditions, and parking schemes. The literature review is structured into distinct methodologies, each addressing specific aspects of smart parking management.

A. RFID based Parking System

Using Radio Frequency Identification (RFID) technology, devices can instantly read information encoded in tags. Consisting of an antenna and microprocessor, RFID facilitates tracking and identification of tags attached to vehicles. This system employs RFID readers, antennas, and tags for managing various operations such as collecting vehicle reports and monitoring parking occupancies [4]. Software enables efficient check-in and check-out of vehicles, reducing congestion near parking lots. By utilizing RFID tags, waiting times at parking gates are minimized, allowing only tagged vehicles to park and enhancing security while preventing unauthorized access.

The RFID system utilizes multiple technologies for automatic vehicle identification and data capture. Generally, the components that are used in the RFID system are RFID tags, RFID readers, and antenna. An antenna is primarily responsible for transmitting the data to the RFID reader and storing the data in the database system. Additionally, a local host computer system is essentially required for the RFID system. The RFID reader picks up the signal from the tag and reformats it so that it can be read. Tag data is transmitted via signal interference once it has been observed. To accomplish this, a database is connected to the host computer system in order to store and preserve the data. In this research, a central database is used to control parking lot input and output.

Anusooya G et. al. [5] suggested transferring the real-time parking occupancy information to the website, mobile application, and the telegram bot app using raspberry pi communicating with the google app engine.

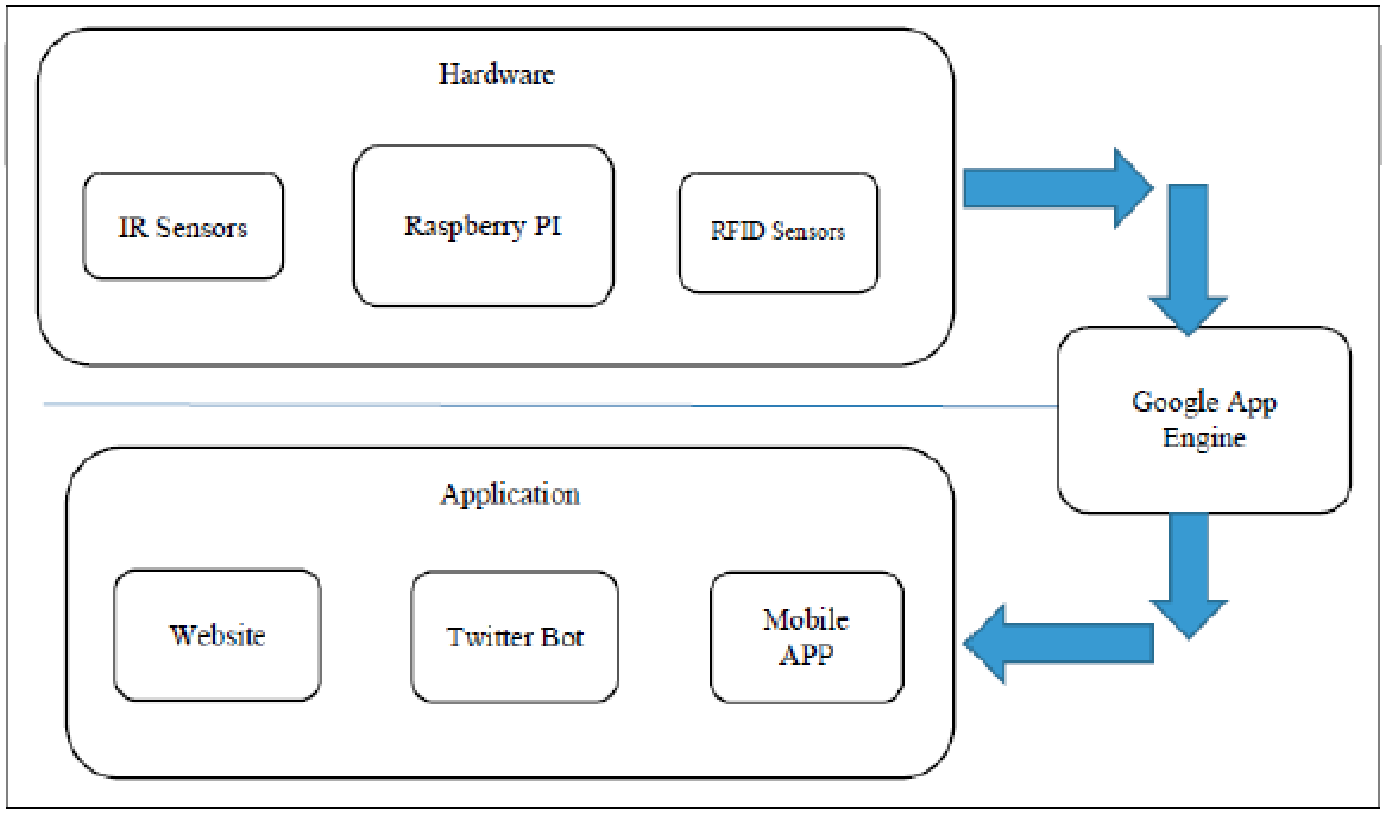
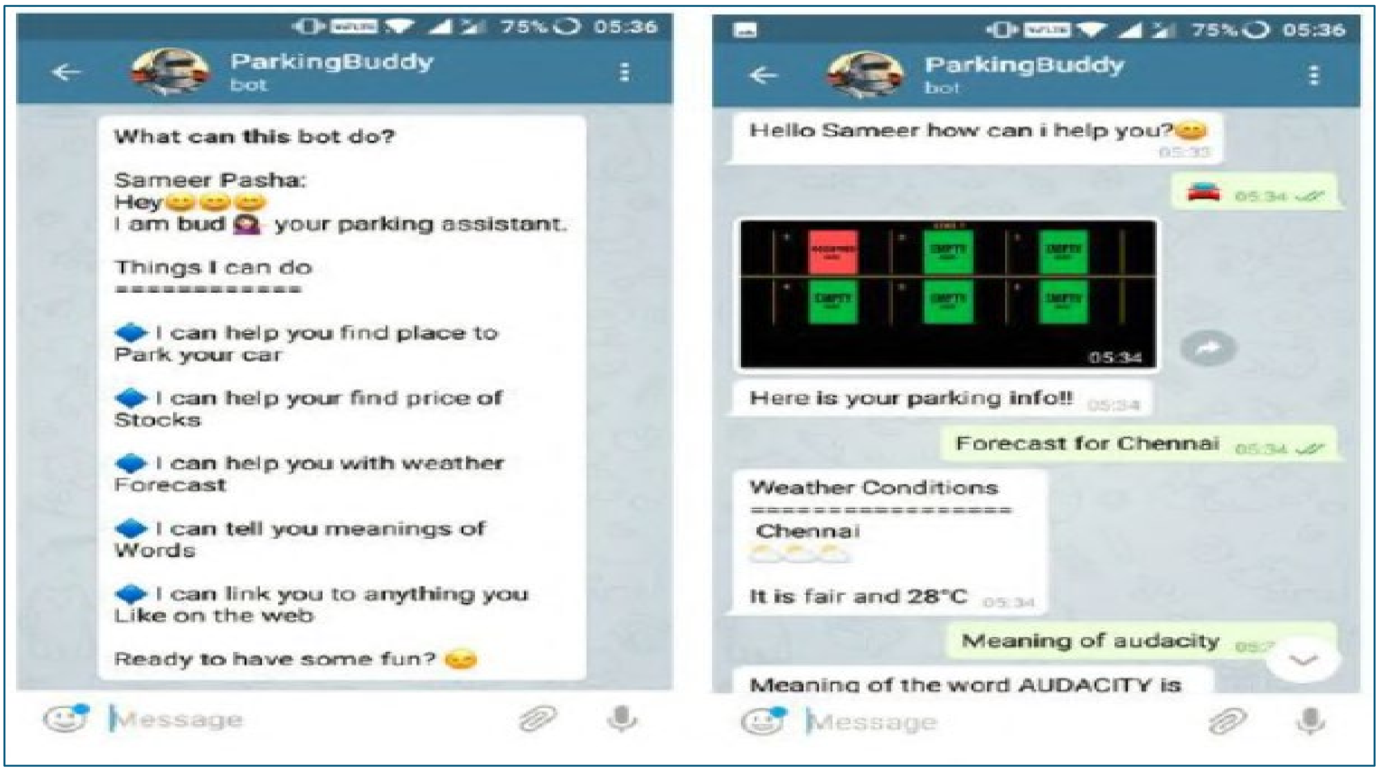


Fig 1: Structure of RFID based Parking System

➢ *The Parking Lot Occupancy Status is Shown on the Different Platforms as:*



ParkingBuddy
bot

What can this bot do?

Sameer Pasha:
Hey😊😊😊
I am bud your parking assistant.

Things I can do
============

🔷 I can help you find place to Park your car

🔷 I can help your find price of Stocks

🔷 I can help you with weather Forecast

🔷 I can tell you meanings of Words

🔷 I can link you to anything you Like on the web

Ready to have some fun? 😊

Message

ParkingBuddy
bot

Hello Sameer how can i help you?😊 05:33

🚗 05:34

05:34

Here is your parking info!! 05:34

Forecast for Chennai 05:34

Weather Conditions
================
Chennai

It is fair and 28°C 05:34

Meaning of audacity

Meaning of the word AUDACITY is

Message

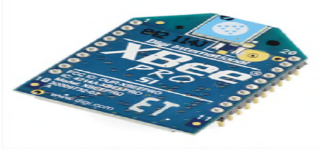
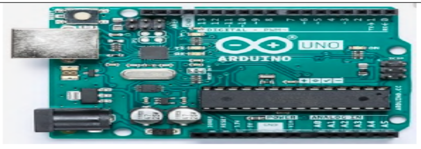
Fig 2: Parking Occupancy Status on Telegram Bot

B. *IoT Based Smart Parking Solution*

IoT has popularized in Smart Cities, addressing issues like traffic congestion, road safety, and smart parking solutions. An IoT-based cloud-integrated system was proposed to develop an automated parking solution, enabling users to locate and book parking spaces via mobile devices before entering the lot [6]. IoT connects physical objects through the internet, allowing them to behave in lifelike manners by performing computation, sensing, and communication. Cloud computing serves as the platform for hosting IoT applications, forming a "Cloud of Things" (CoT) for remote monitoring, accessing, and controlling objects. This system can be implemented for indoor and outdoor parking systems using sensors like Geomagnetic, Fiber Bragg Grating, and Ultrasonic sensors, with Geomagnetic sensors wirelessly communicating parked vehicle status by sensing changes in the magnetic field [7].

The hardware and their uses in IoT based parking management systems are shown in table –

Table 1: Hardware and their Uses

Hardware Image	Hardware Name	Hardware Level
	Ultrasonic Sensor	Sensor Level
	XBEE (Series 2)	Display Level
	Arduino Uno	Programming Level
	Arduino Mega	Programming Level

Generally, an ultrasonic sensor is used for outdoor parking systems because they are cost effective and come with better accuracy. XBee sensors provide a large range of signal transfer distances (40 m indoor capacities and 120 m outdoor capacities). The Uno module provides better memory management and is compatible with the XBee radio. Light emitting diodes (LEDs) are used because of their visibility efficiency from every direction and mostly for their power efficiency.

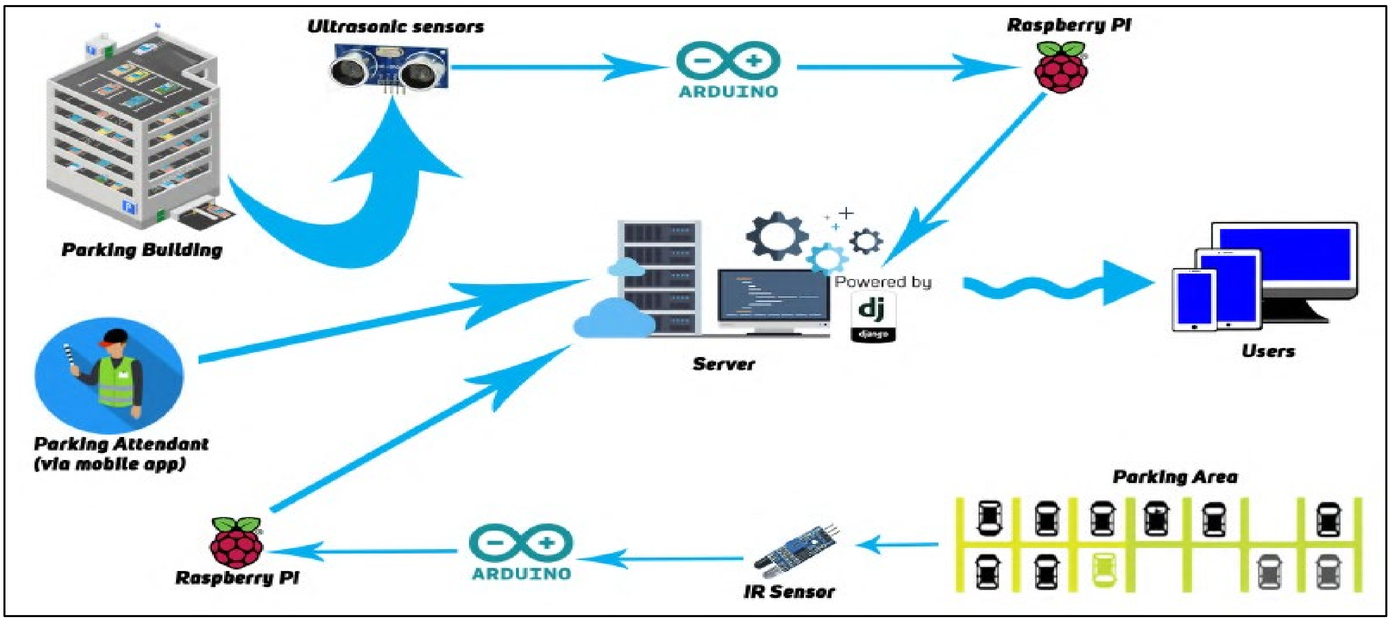


Fig 3: System Architecture and Working Procedure

This parking solution is time effective and worked well with great efficiency with some minor errors like double parking issues and improper parking vehicle identification. These errors can be resolved using human interference or using some more modern technologies. A big parking lot may require a lot of sensors, and regular checks and maintenance of the sensors are also required at some periodic intervals, hence the system is not cost-effective for every place.

C. *Smart Parking Solution with Image Processing*

A camera is required as the main hardware component, which makes this system very inexpensive and simple to obtain. The camera is used to locate the free space in the parking lot in real-time by image sequence gathering and detection of free space. The subtraction method is applied to identify moving vehicles. A software component of the system could be an image-processing program such as MATLAB or Python [8]. MATLAB software is used to detect vehicles in the parking lot using the input from the camera in real time. After picking up the image, it is processed through multiple image processing commands to understand the image more efficiently, and output for the same is obtained. The block diagram is shown below as-

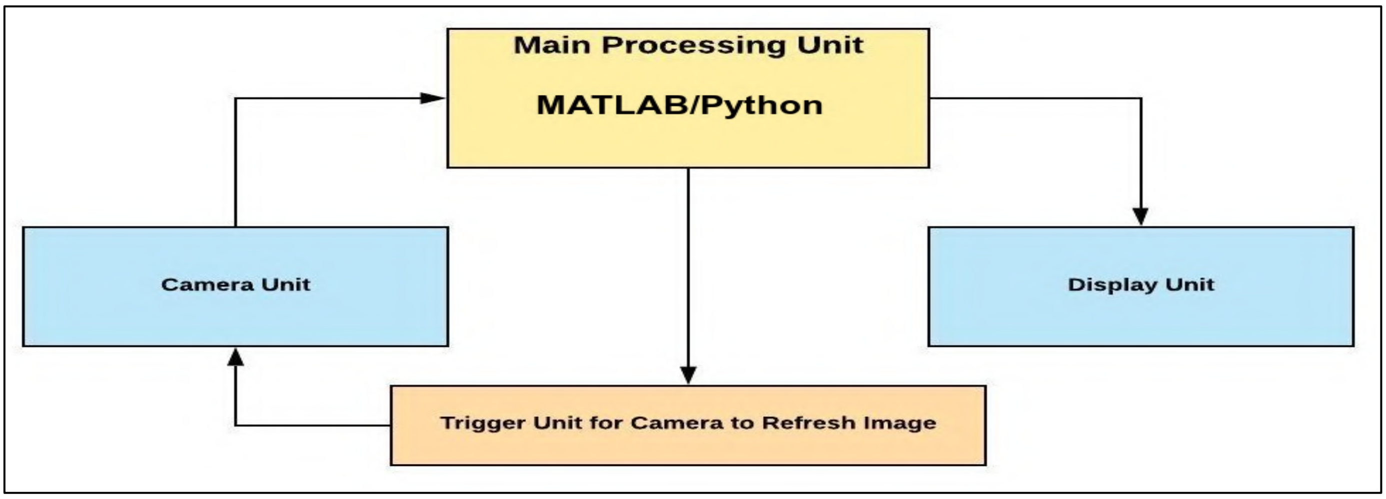


Fig 4: Flow Structure of Image Processing

When the image is captured from the camera, image processing is done to mean the act of arbitrarily altering an image to conform to an aesthetic standard or to provide evidence for a preferred reality, and noise reduction is generally accomplished by an image filtering technique. The reference image is taken initially, and the captured image is subtracted from the reference image to check the changes and similarities between the two before further processing is done.

For the removal of generated noise, morphological processes, i.e., dilation and erosion, are employed. The image is converted into a grey image, and image enhancement techniques are used. Then, corners are identified and compared by using edge detectors to check for similarities and differences. The whole system can be divided into the following sequential manner-System Initialization, Image Acquisition, Image Segmentation, Image Enhancement, Image Detection.

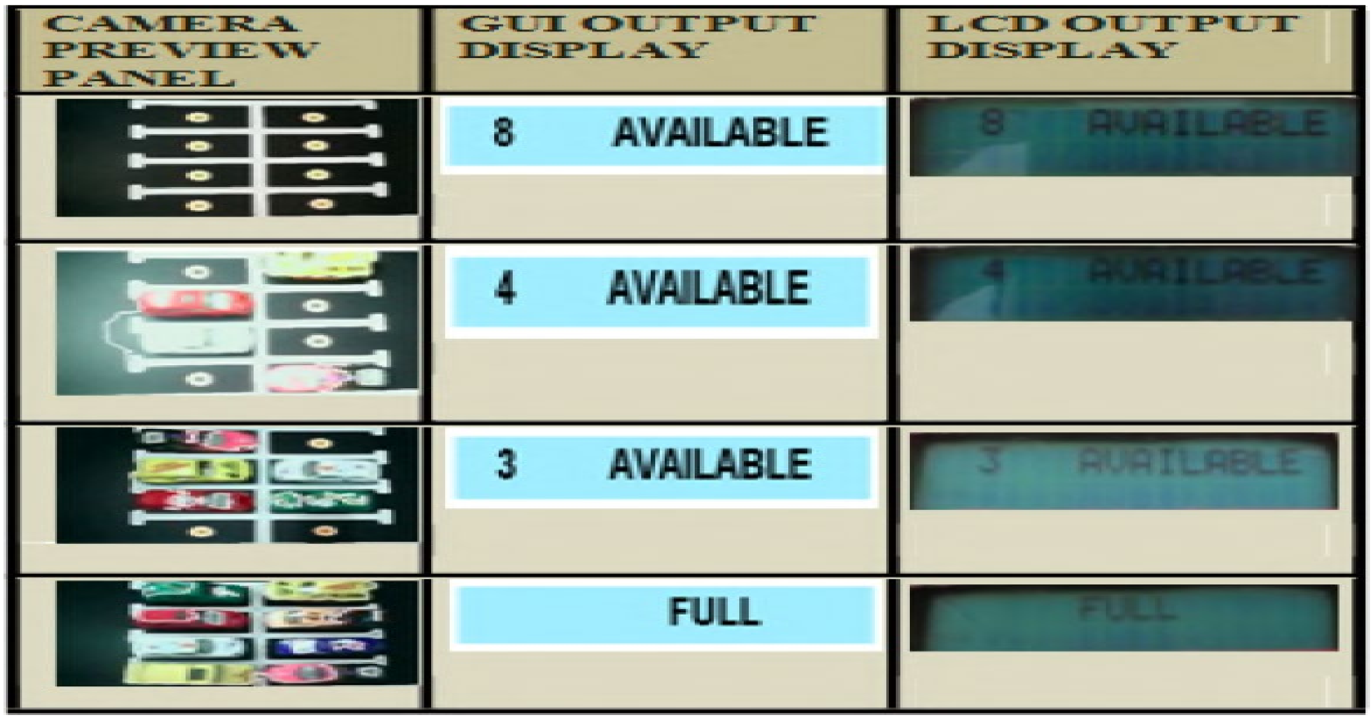


Fig 5: Parking Slot Availability Result on Display

Canny edge detection is used by J. Trivedi et al. [9] and Katy Blumer et al. [10] to create a real-time intelligent parking management system. A USB camera was used to get the input video feed, LCD board showing current parking lot capacity, and a Raspberry Pi module to transfer the information using Wireless Fidelity (Wi-Fi).

Although this method is cost-effective, it has a limitation in that it's most effective for smaller parking areas. It may lead to false results in cases when some different objects are present instead of cars.

D. Machine Vision based Automated Parking System

Machine learning-based technology has been extensively researched in recent years due to its flexibility and cost-effectiveness. Computer Vision is the subset of 'artificial intelligence' (AI) technologies that enable computers and systems to generate relevant information from visual inputs such as digital images and videos.

Growth in AI and innovation in neural networks and deep learning led to machines surpassing humans in various tasks related to labelling and detecting objects. The production of massive amounts of data is the primary driver behind the acceleration of computer vision technology, which is used to train the model and make the model robust. In the context of smart parking solutions, Computer vision technique-based methods have been tested to overcome challenges as mentioned- a.) Marking and detection of parking slots. b.) Classification of the parking slot based on the occupancy of parking spaces. c.) Detection and counting the number of vehicles present in the parking lot [11]. Magnetometers and ultrasonic sensors are no longer necessary because a single camera can monitor a larger area. Furthermore, the camera's installation and maintenance costs are less, and it can help with extra tasks like detecting theft and looking into unusual driver behavior [12, 13].

Numerous publicly accessible datasets are accessible for advancing the research on computer vision-based parking management systems. Some of them are as- PKLot Dataset [14], CNRPark, CNRPark-EXT, PLDs dataset [15,16,17].

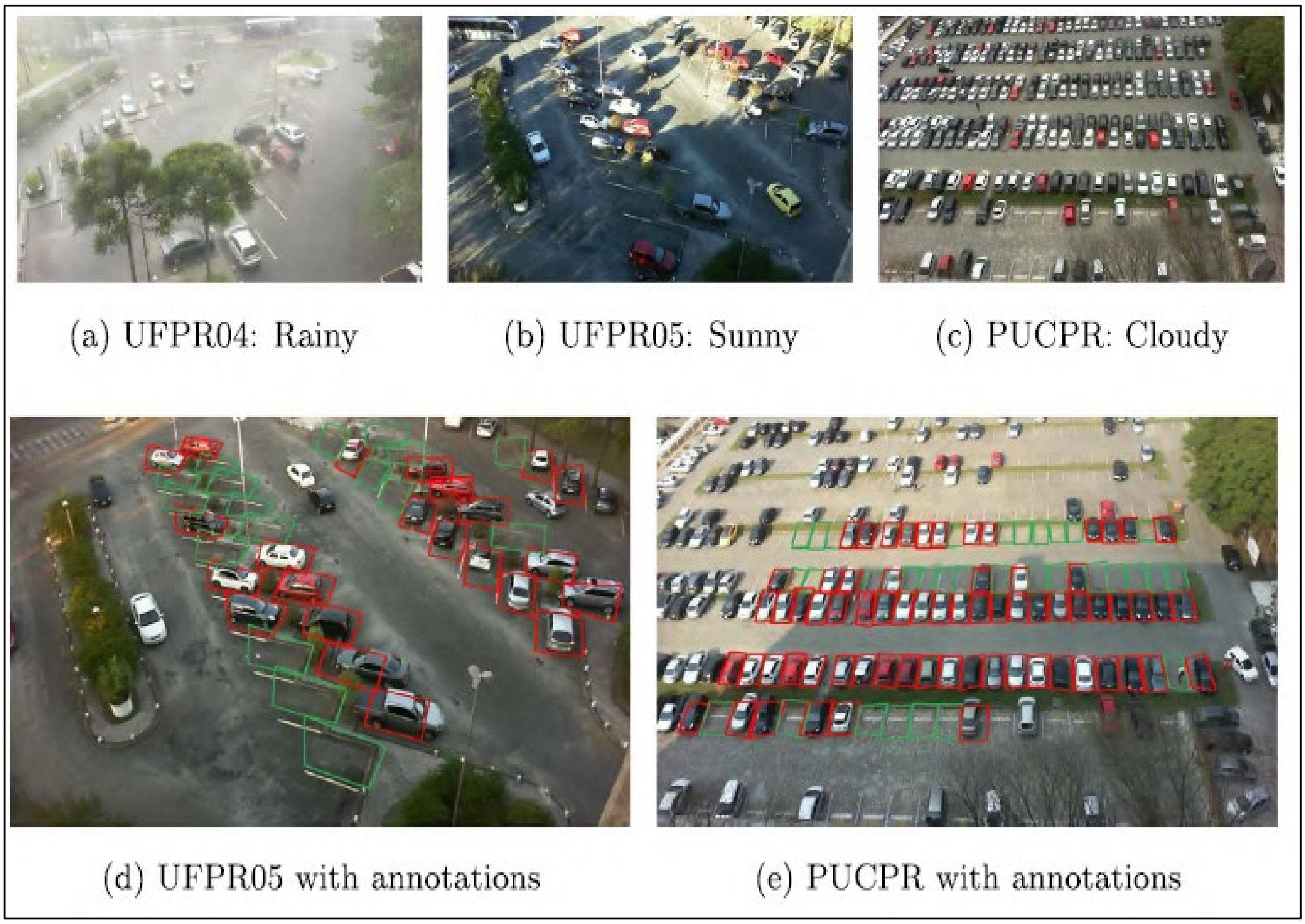


Fig 6: PKLot Dataset: Red Box Shows Occupied Parking Space while Green Box Shows Unoccupied Parking Space

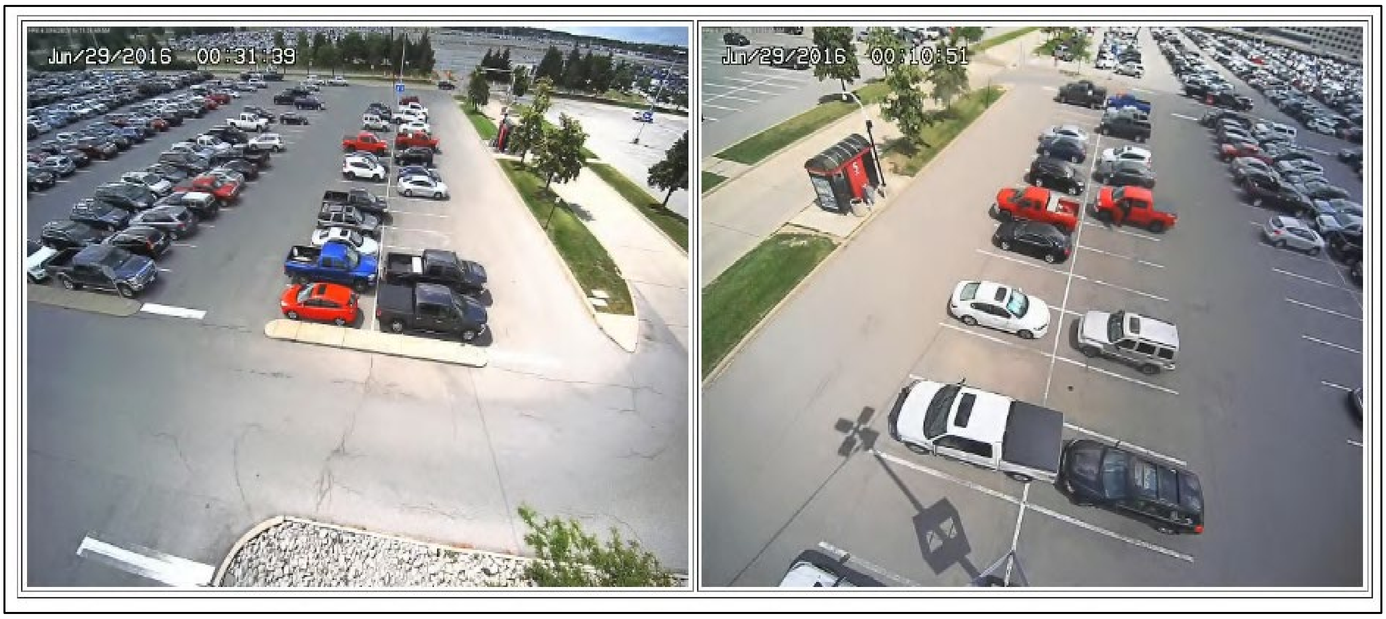


Fig 7: Parking Lot Dataset (PLDs)

Furthermore, to determine whether a parking space is occupied or available for use, each parking space classification task can be viewed as a binary problem. These methods can be classified into 2 major categories-

- Feature extraction-based methods
- Deep learning-based methods

➢ *Feature Extraction-Based Methods:*

All the images acquired by the camera are segmented into individual parking spaces as a pre-processing step. The image is typically scaled, and histogram equalization tasks are performed to make it more suitable for feature extraction. In the feature extraction step, one or more feature vectors are extracted from the images, such as Local Phase Quantization (LPR), Local Binary Pattern (LBP) and Histogram of oriented gradient (HOG) [18]. Then a classifier (support vector machine (SVM), multilayer perception (MLP), etc.) is used for training the model and feature vectors from the images and the ground truth of each image is fed to train the classifier. Then this trained model is used to perform predictions based on unseen images. Almeida et al. proposed to use Local Phase Quantization (LPQ) and LBP as feature vectors and SVM as the classifier. Later, Almeida et al. used LPQ and LBP as feature descriptors and an ensemble of SVMs was used as a classifier on the PKLot dataset. In Suwignyo et al. [19], LBP has been used as a feature extractor and KNN and SVMs are tested as a classifier for the parking management solution. In Dizon et al., as feature descriptors, LBP and HOG were used, and a linear SVM classifier was employed.

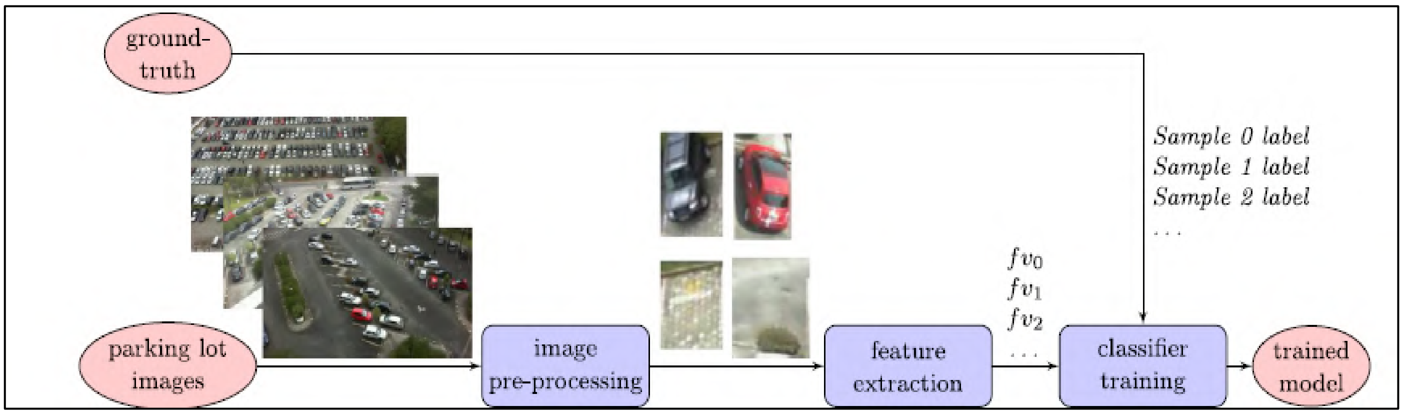


Fig 8: High-Level Scheme of Feature Extraction Based Methods

➢ *Deep Learning-Based Methods:*

Deep learning-based methods have a similar workflow as the feature-based approach, except for the feature extraction and classifier treating separately, these 2 steps are combined as a representation learning block and can be divided into (a.) Transfer learning on pre-existing CNNs for classification, such as LeNet, AlexNet, VGG and ResNet [20]. (b.) Custom CNN generation based on pre-existing Convolutional Neural Networks (CNN)s (c.) Use of deep learning-based Object Detection methods, such as- Fast-RCNN, Faster-RCNN, Mask-RCNN, Single Shot Detector (SSD) and You Only Look Once (YOLO) etc. Yoshinki et al. [21] used the concept of transfer learning of the neural network, it was first trained on a generic dataset before fine-tuning for a parking lot dataset. Because of the compactness of the network structures of AlexNet and LeNet, these are used by Julien Nyambal et al. for the parking lot management system.

Ding and Yang et al. [22] proposed a YOLOv3 with added residual blocks to extract more granular features from the images. It is employed to classify pictures of parking lots. Microsoft Common Objects in Context (COCO) and PASCAL Visual Object Classes Challenge (PASCAL VOC) datasets were used to train the model and then fine-tuned. Based on CNN architectures of the AlexNet, LeNet and VGGNet, some lightweight models are also proposed. These models are primarily created for systems with low processing power, like smart cameras, and tested on the PKLot dataset. Merzoug et al. [23] used a lightweight network to classify the parking spaces by the MobileNetV2 algorithm and tested it for PKLot, CNRPark-EXT, and a private parking dataset.

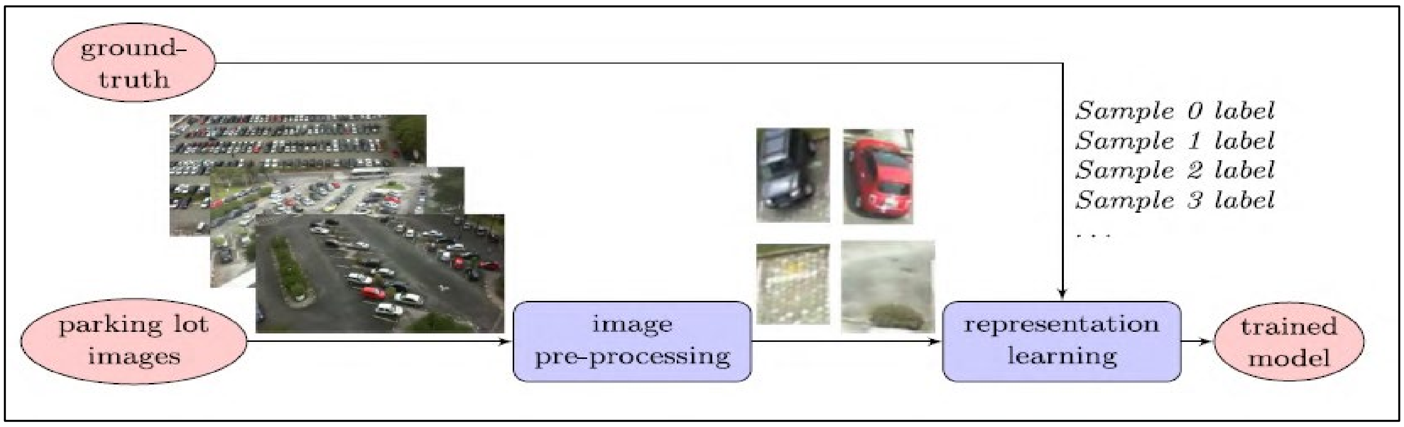


Fig 9: DL-Based Methods High-Level Scheme

III. THEORETICAL BACKGROUND

Object detection is the basic research platform in the field of computer vision, artificial intelligence, deep learning, etc. It aims to locate and identify objects of various categories, quickly and accurately in each image. It has been widely used in image and video retrieval, autonomous driving, intelligence surveillance system, medical image analysis, etc.

There are several kinds of object detection queries, which can be understood through the image below:

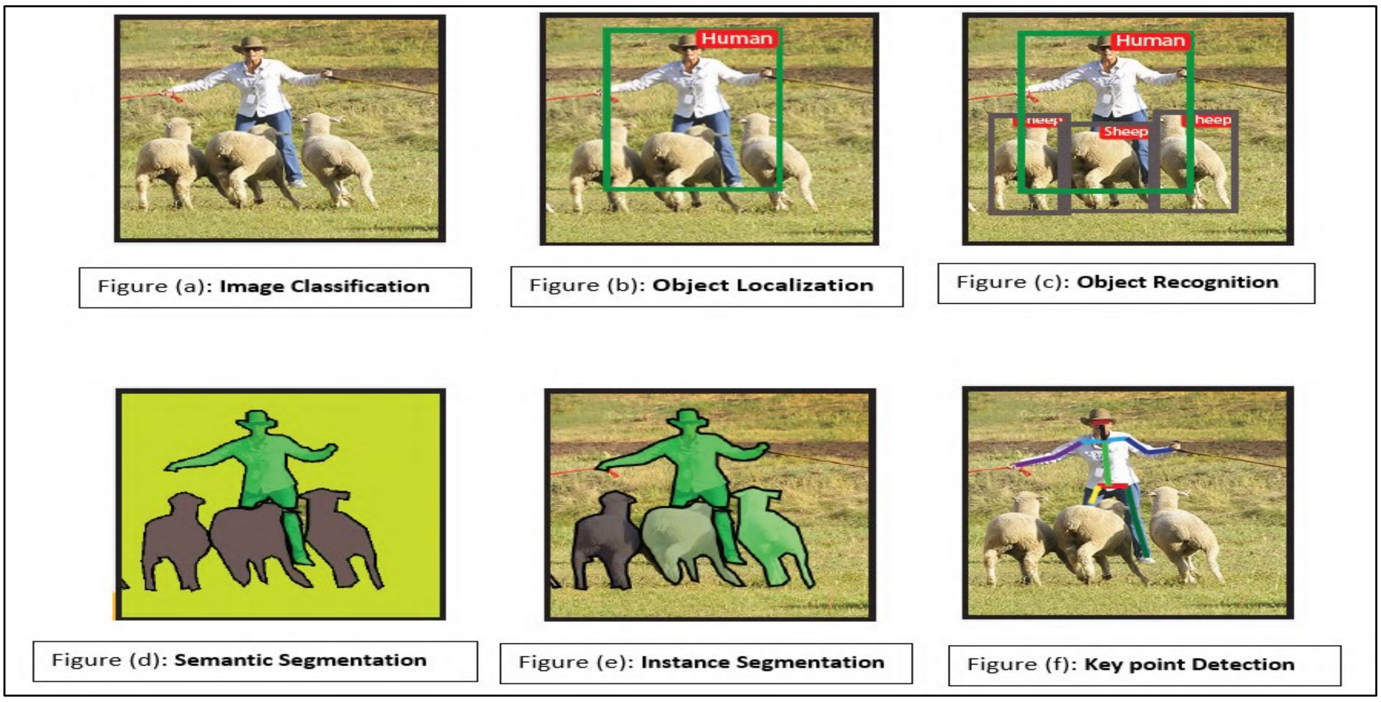

Human
Sheep
Figure (a): Image Classification
Figure (b): Object Localization
Figure (c): Object Recognition
Figure (d): Semantic Segmentation
Figure (e): Instance Segmentation
Figure (f): Key point Detection

Fig 10: Different forms of object Detection in Deep Learning.

The image classification model is evaluated and examined based on mean classification error. The best-performing model is chosen based on "precision" and "recall" across every possible best-matching bounding box for the known objects in the ground truth image [24].

Following is a popular example of object detection algorithms for better understanding and to demonstrate how an object detection algorithm looks like:

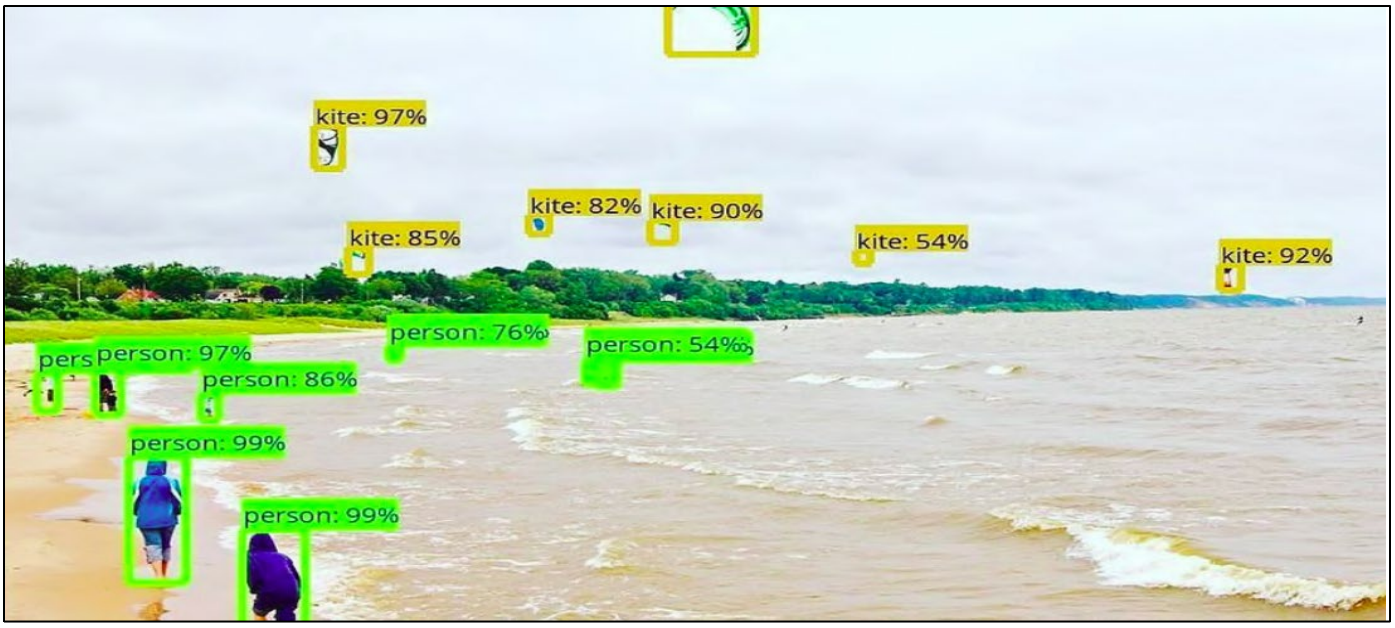


Fig 11: Object Detection with Multiple Classes

To understand the basic deep learning approach, for detecting multiple class images the knowledge of CNNs is a must [25].

Convolutional Neural Networks (CNNs) are advanced artificial intelligence systems designed to process visual information, inspired by the human brain's ability to recognize objects in images. These networks are particularly adept at tasks like image recognition, object detection, and medical diagnosis.

CNNs operate by breaking down images into smaller, more manageable components, similar to how our brains analyze visual information. They do this through layers of convolutional and pooling operations. Convolutional layers use filters to scan images for patterns, such as edges and textures, creating feature maps that highlight these features. Pooling layers then reduce the size of these feature maps, focusing on the most relevant information while discarding unnecessary details.

Following convolution and pooling, CNNs pass the feature maps through fully connected layers, where the network interprets the features to make predictions. For example, if trained to recognize cats, the fully connected layers analyze the features and determine if they resemble cat patterns.

CNNs are trained through backpropagation, where the network adjusts its parameters based on the difference between its predictions and the true labels of the images it's trained on. This iterative process allows CNNs to learn from experience and improve their accuracy over time.

CNNs are sophisticated algorithms that mimic human visual processing. By analyzing images in layers and recognizing patterns, they can accurately identify objects and features within images, making them valuable tools for various applications in fields like computer vision, healthcare, and autonomous vehicles.

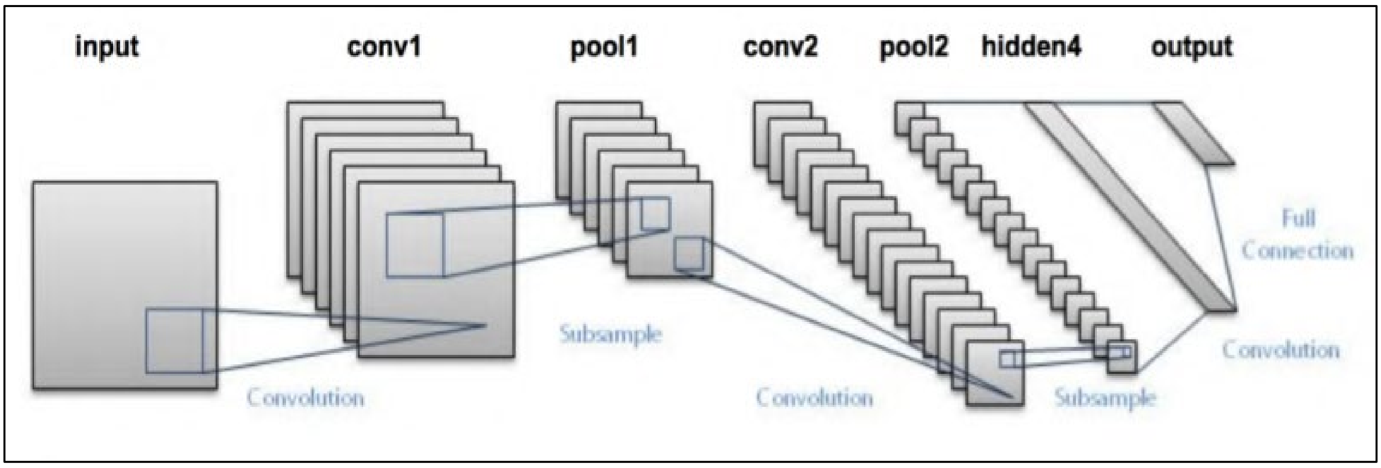


Fig 12: CNN Workfow Structure [26]

A. *Transfer Learning Paradigm and Fine-Tuning*

Transfer learning is a machine learning method, where a model is developed for a task is reused as a starting point for a model on another task.

In deep learning, pre-trained models are frequently used as starting points for computer vision tasks because they require less time and computation than developing neural networks. In this, pre-trained weights are used, so the training doesn't need to start from scratch [27].

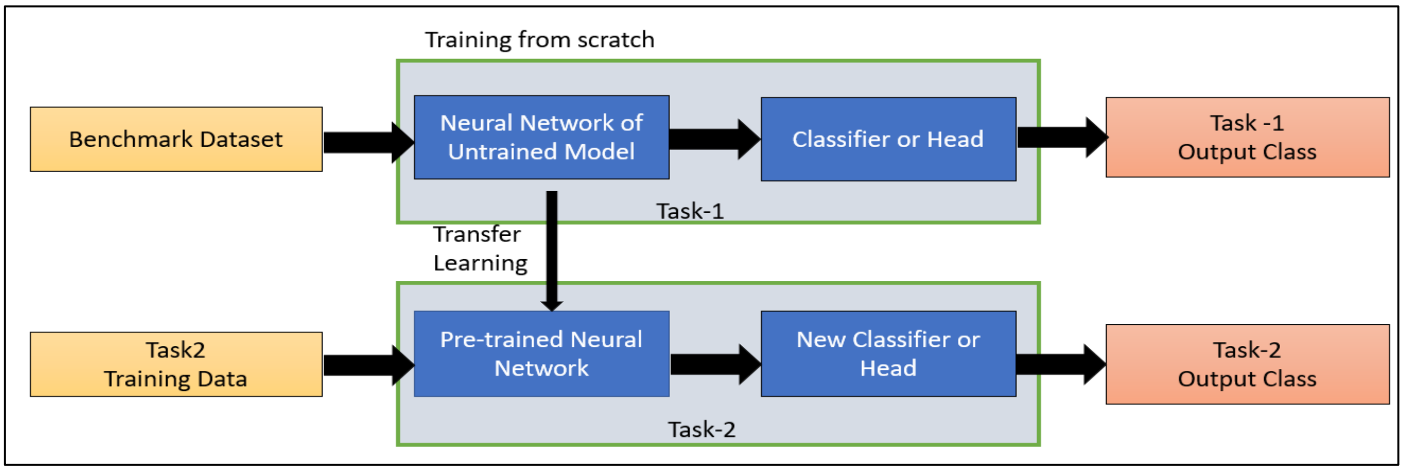


Fig 13: Systematic Illustrating of Transfer Learning Paradigm

In a specific manner, it is not a methodology but just a paradigm used actively in computer vision. The training pattern obtained in transfer learning is called "fine-tuning". After fine-tuning, the parameters are reduced to a significant level, which ultimately reduces the training time required. Training from scratch requires a large dataset, while transfer-learning is useful when there is a small dataset available. The only constraint while using transfer learning is to ensure that the input to the network is similar or identical.

The objective of the training phase of all the neural networks is to reduce the value of the loss function. Loss functions are generally defined as the statistical measure of the model performance based on accurate prediction. Classification loss (cross entropy loss and focal loss) and Regression loss functions (mean square error, mean absolute error, hubler loss functions) are being used for classification and regression task respectively for neural network.

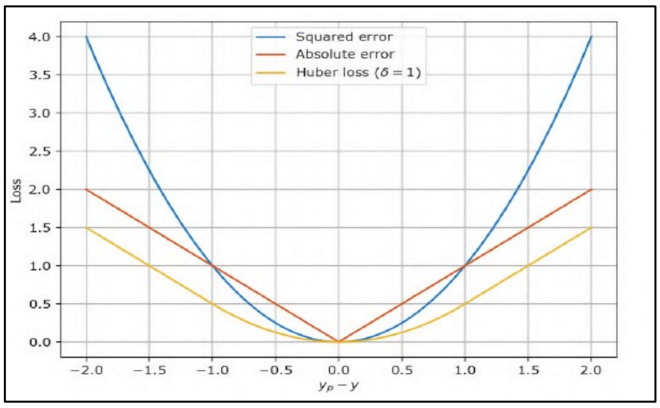


Fig 14: MSE (blue), MAE (red) and Hubler Loss (yellow)

B. Basic CNN Architectures

In deep learning, several types of Neural Networks are used in image processing, image classification, segmentation, self-driving cars, etc. range of applications. One such popular neural network is the convolutional neural network (CNN).

CNN, or ConvNets, are the multi-layer NN architecture that is used to discern visual patterns from the image grid cells. CNN consists of numerous layers, such as convolutional layers, pooling layers, and fully connected layers, and utilizes the backpropagation algorithm to obtain the spatial hierarchy of the features present in the image adaptively and automatically. The summary table of various CNN architecture models is below:

Table 2: ISLVRC Competition CNN Architecture Models

Author & Year	CNN	Error rate	Parameters
Yan Lecum et al. (1998)	LeNet	—-	60 Thousands
Alex Krizhevsky et al. (2012)	AlexNet	15.3%	60 Million
Mattew Zeller et al. (2013)	ZFNet	14.8%	58 Million
Google (2014)	GoogleNet	6.67%	4 Million
Simyonyan et al. (2014)	VGGNet	7.3%	138 Million
Kalming He et al. (2015)	ResNet	3.60%	23.4 Million

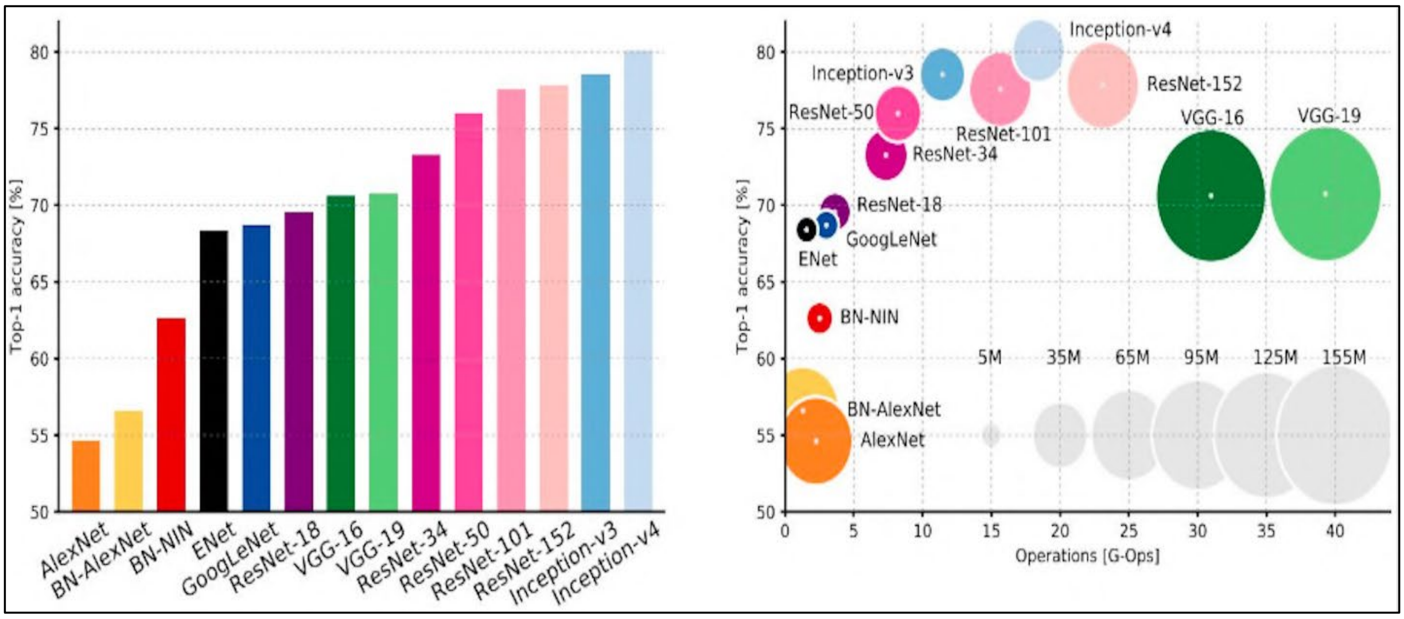


Fig 15: Comparison of Various Kinds of CNN Architectures in Terms of Accuracy

C. Object Detection Algorithms

After understanding the concepts of various CNN algorithms, it becomes crucial to have knowledge about object detection algorithms. CNNs are able to perform classification tasks, but it is also recommended to estimate the location of the object present in the input image. This whole task is referred to as 'Object Detection'.

Object detection algorithms are being used for various applications, such as face detection, pedestrian detection, vehicle detection, medical image analysis, etc. However, because of significant variations in poses, occlusion, viewpoints, and lighting conditions, it is becoming challenging to accomplish object detection perfectly with additional object localization tasks. In recent years, Object Detection applications have attracted a lot of attention.

Before the development of CNN architectures, there were some additional object detection algorithms. These methods usually consist of 3 different stages. The first stage is associated with framing the candidate region on the input image. For this purpose, a sliding window approach is used. The second stage is associated with feature extraction from the selected candidate region using 'Scale-Invariant Feature Transform' (SIFT) and HOG algorithms. The third stage is associated with the classification task, whose main objective is to predict the class label of the selected objects. The classifiers used for such purposes are obtained from machine learning, i.e., 'support vector machine' (SVM) and AdaBoost. algorithms etc. Classification accuracy is basically measured by calculating the precision and recall estimates. 'HOG + SVM' and 'HOG + Cascade' combinations have been used for such purposes.

Based on DNNs, modern object detection algorithms have been classified into the following 2 categories-

- Two-stage detection algorithms
- Single-stage detection algorithms

Two-stage object detection algorithms are used in computer vision to identify objects in images or videos. They consist of two main stages: region proposal and object classification. In the region proposal stage, potential object regions are generated using techniques like selective search or edge boxes. These regions are then passed to the object classification stage. Here, the algorithm determines the presence of objects and assigns them to specific classes using deep learning methods like Convolutional Neural Networks (CNNs). Object classification involves analyzing the content of each proposed region and assigning probability scores to different object classes. After classification, the algorithm refines the bounding boxes of detected objects to improve localization accuracy. Techniques like non-maximum suppression are used to remove redundant bounding boxes and select the most likely object detections based on their confidence scores. Overall, two-stage object detection algorithms are powerful tools for accurately identifying and localizing objects in complex scenes, with applications in autonomous driving, surveillance, and video object tracking.

RCNN, Fast RCNN, Faster RCNN [28] and Mask RCNN [29] are the popular algorithms as 2 stage object detection algorithms.

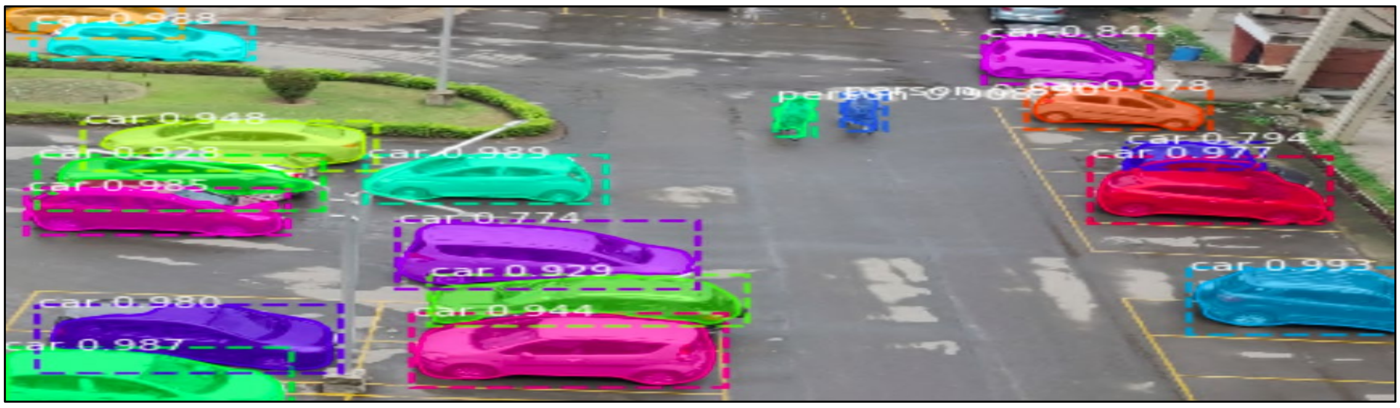


Fig 16: Output of Mask R-CNN Algorithm at Parking Area

Single-stage object detection algorithms are a class of computer vision models designed to detect objects within images or videos in a single pass without the need for region proposal. These algorithms are efficient and fast, making them suitable for real-time applications. Unlike two-stage algorithms, which typically involve region proposal followed by object classification, single-stage algorithms directly predict object bounding boxes and class labels from input images.

One popular single-stage object detection algorithm is YOLO (You Only Look Once). YOLO [30] divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell. This approach allows YOLO to detect multiple objects in a single forward pass, making it highly efficient.

Another example is SSD (Single Shot MultiBox Detector) [31], which uses a similar approach to YOLO but introduces multiple feature maps at different scales to detect objects of various sizes. SSD achieves high accuracy by combining information from multiple feature maps at different resolutions.

Overall, single-stage object detection algorithms offer a balance between speed and accuracy, making them well-suited for real-time applications where fast inference is crucial. They continue to be an active area of research in the field of computer vision, with ongoing efforts to improve their efficiency and accuracy.

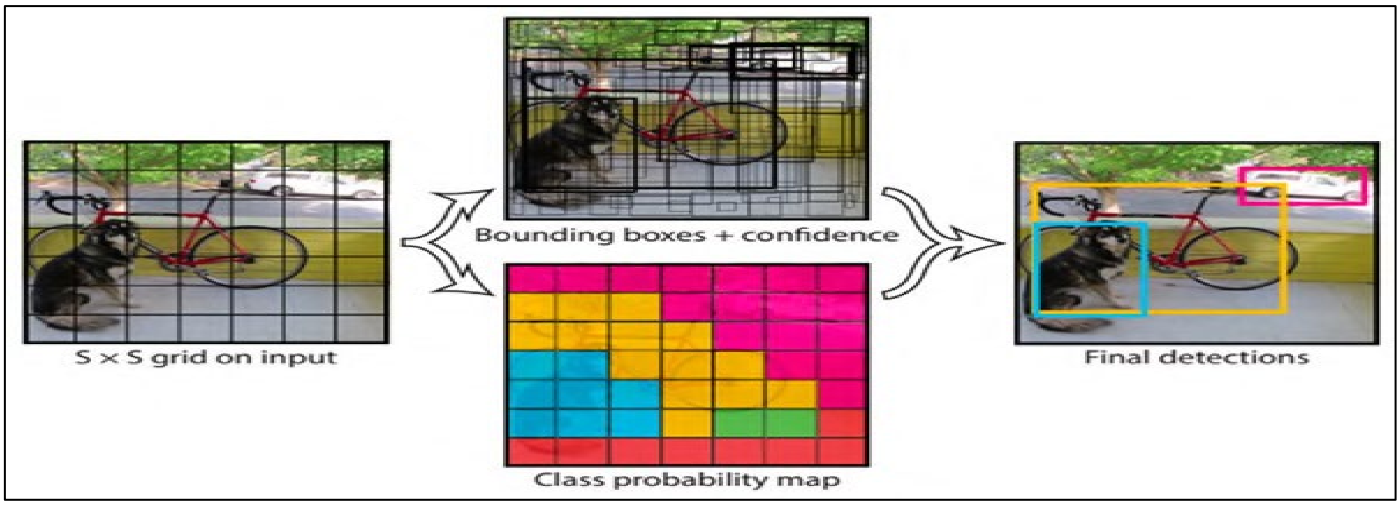
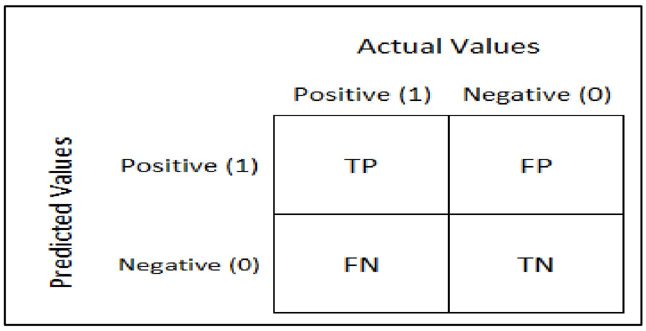


Fig 17: A Simplified Illustration of the YOLO Object Detection Pipeline

D. Accuracy of Object Detection Model

By comparing the detected objects from the detected objects to ground reference data, the accuracy matrices for object detection algorithms define the accuracy of a deep learning model. These accuracy matrices mainly include the following matrices- Confusion matrix, F1 score, Precision-Recall curve and COCO 'mean Average Precision' (mAP).



		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Fig 18: Confusion Matrix

- **Precision:** Precision is defined as the proportion of true positives to all positive predictions.

$$precision = \frac{TP}{\text{TP+FP}} \quad (1)$$

- **Recall:** Recall is the proportion of true positives to actual (relevant) objects

$$recall = \frac{TP}{\text{TP+FN}} \quad (2)$$

- **F1 Score**: F1 score is de_ned as the weighted average of precision and recall values. The range of its values is 0 to 1, with 0 denoting the lowest accuracy. Alternatively, the F1 score is the measure to evaluate the balance between precision and recall.

$$F1\ Score = \frac{2 * Precision}{\text{Precision+Recall}} \quad (3)$$

- **Precision-Recall Curve**: The performance of an object detection model is assessed using this plot of precision (y-axis) and recall (x-axis). The area under this curve is called AUC.

- **Average Precision**: The average precision (AP), which represents the average of all the precision for a class object tested for different IoU thresholds, is the compressed way to represent the AUC. Mathematically, AP is defined as-

$$\text{AP} = \sum_{k=0}^{n-1}[recalls(k) - recalls(k+1)] * Precision(k) \quad (4)$$

- **Mean Average Precision**: Depending on the various detection challenges that are present, the mean Average Precision, or mAP score, is calculated by taking the mean AP across all classes and/or overall IoU thresholds. Mathematically, mAP is defined as below-

$$mAP = \frac{1}{n\sum_{i=0}^{n} AP(\,i)} \quad (5)$$

The performance of various computer vision models can be compared using mAP. mAP provides researchers and engineers working in computer vision with a single primary metric that considers both precision and recall when evaluating models.

IV. SETUP AND METHODOLOGY

All the design parameters are looked at, and a preliminary set-up for an experiment and its challenges are described-

A. Setup of Camera Sensor:

As a result of network and hardware advancements, desktop computing has been surpassed by the burgeoning mobile computing industry, in which smartphones, tablets, and computers play a significant role in people's lives. It also features several sensors such as microphone, gyroscope, accelerometer, digital camera, digital compass, GPS, enabling the development of apps to help with daily tasks.

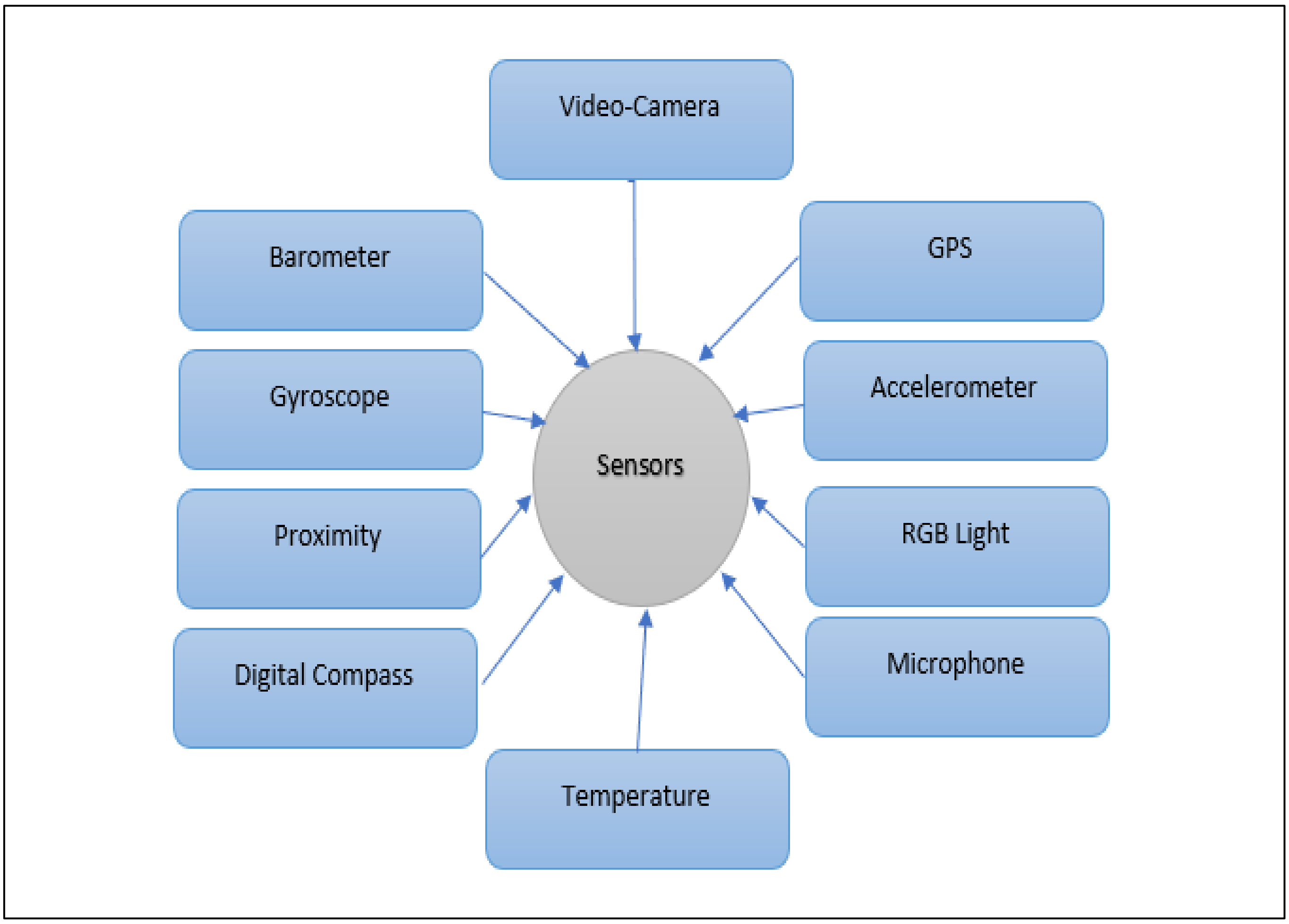


Fig 19: Sensors Embedded in a Smartphone [32]

B. Data Collection

For the project, a mobile camera (Redmi Note 10 Pro) was used to record video of the parking area, which was mounted on a tripod stand and placed on the two-story building near the parking lot. The captured area belongs to the travel companies associated with the 'Indian Institute of Technology', Kanpur (IIT Kanpur). The height and orientation of the tripod stand were kept constant to _x the frame size of the video. The video data is collected for 4 hours from 11 a.m. to 3 p.m. each day from Wednesday, 04/01/2033, to Tuesday, 17/01/2023, through a two-story building near IIT Kanpur main gate no. 2. The OpenCamera Android application was used to capture and store the data, and the video resolution was set to a high-definition frame rate (HD) at 1,280 x 720 (16:9, 0.92 MP). The image corresponding to the setup of the camera is shown. A power bank was also attached to the camera to provide a power supply during the whole period of recording.

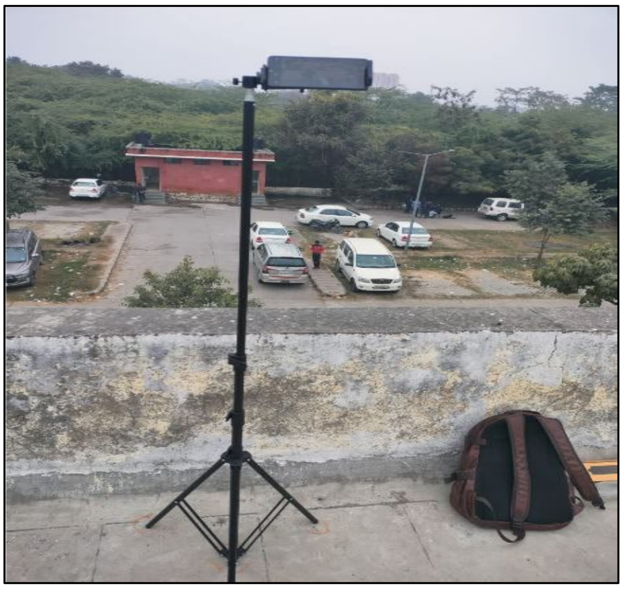


Fig 20: Camera Setup on the Roof of the Building

C. Methodology:

This section includes information about the model architecture, a justification for the model selection, and a detailed mathematical description of all the steps that were taken throughout the entire process.

➤ *Overview of the Process:*

Design and analysis of deep learning computer vision framework that can be tracked with the following steps-

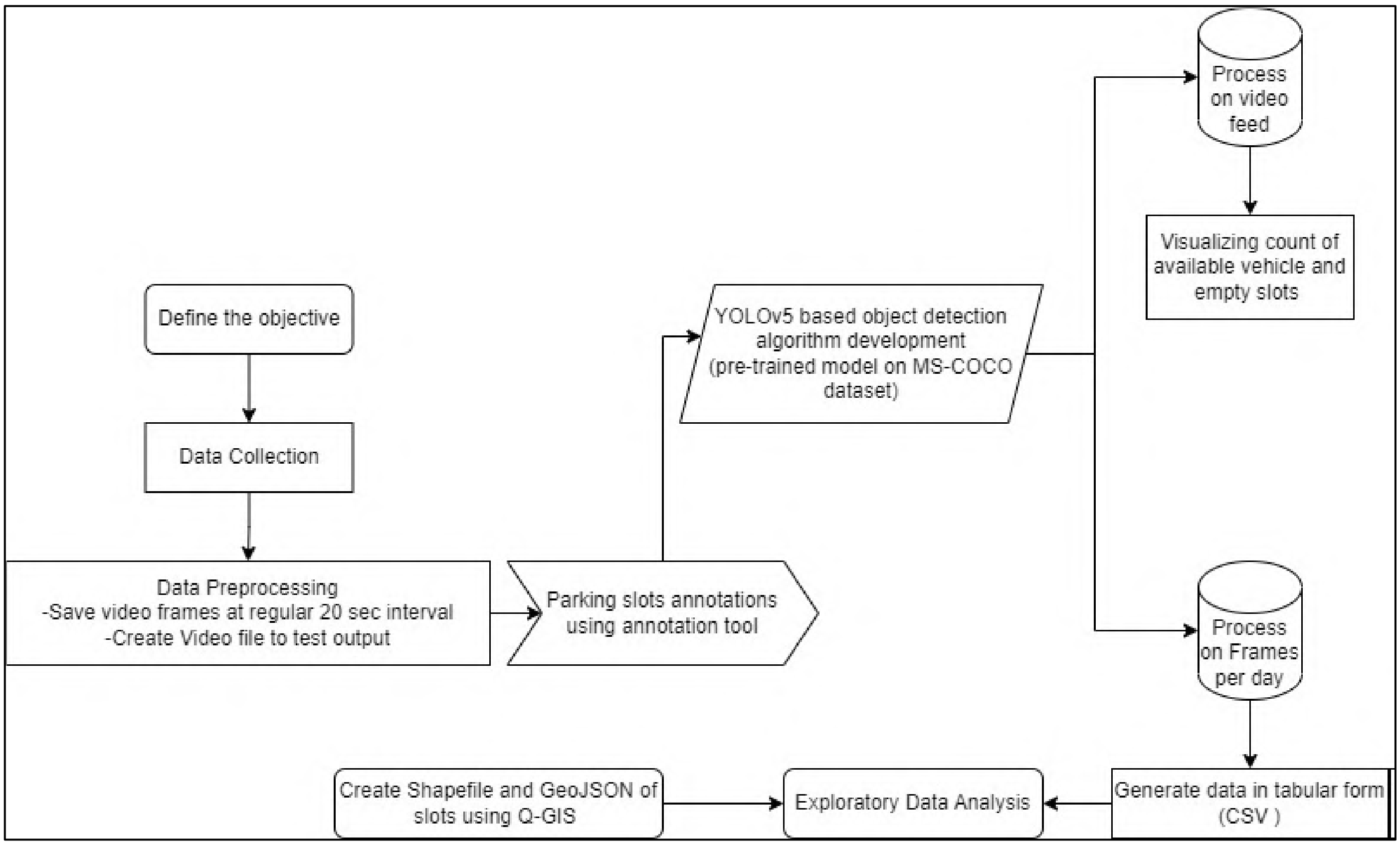

Define the objective
Data Collection
Data Preprocessing
-Save video frames at regular 20 sec interval
-Create Video file to test output
Parking slots annotations using annotation tool
YOLOv5 based object detection algorithm development (pre-trained model on MS-COCO dataset)
Process on video feed
Visualizing count of available vehicle and empty slots
Process on Frames per day
Generate data in tabular form (CSV)
Exploratory Data Analysis
Create Shapefile and GeoJSON of slots using Q-GIS

Fig 21: Overview of Deep Learning-based Object Detection Framework

Data is collected from the roof of the building near IIT Kanpur Gate 2 near the parking lot. This data constitutes the most critical component of any deep learning process.

After collecting the two weeks of data continuously, data pre-processing is required. Video pre-processing is associated with multiple techniques used to analyze, modify, and enhance video data. The process involves tasks such as resizing, color correction, noise reduction, and frame extraction.

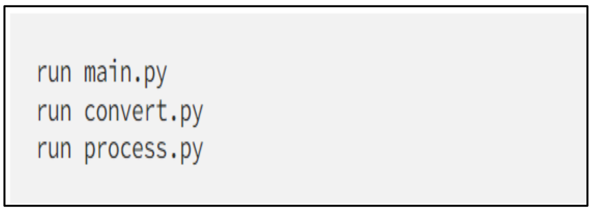
To create computer vision and deep learning applications, image annotation is a crucial first step. The process of annotating images involves labelling regions, objects, and attributes within the images to provide semantic information for the training and evaluation of the algorithms. For the current research, Yolo Annotation Tool was created to annotate the frames using a manual approach.

For the project, mask RCNN and YOLOv5 algorithms were used as the base object detection algorithm, which is quick and effective but might not be as precise as other algorithms when trying to spot smaller or partially obscured objects. Q-GIS was used to perform the geospatial analysis from the parking data and the exploratory data analysis is performed. It is the step involved in the data analysis process and involves using summary statistics and visual representations of the data to look for insights by plotting various graphs, detecting outliers, test hypotheses, and underlying assumptions.

➤ *Marking of the Parking Spaces*

Yolo Annotation Tool is used to annotate the objects in the image or frame to train and test object detection efficiently and quickly in their respective works.

Yolo Annotation Tool is an open-source toolbox for performing above mentioned task on the video frame or image. OpenCV acts as a pre-requisite for this purpose. The steps required to perform annotations are as below-



```
run main.py
run convert.py
run process.py
```

Fig 22: Steps Involved in Yolo Annotation Tool

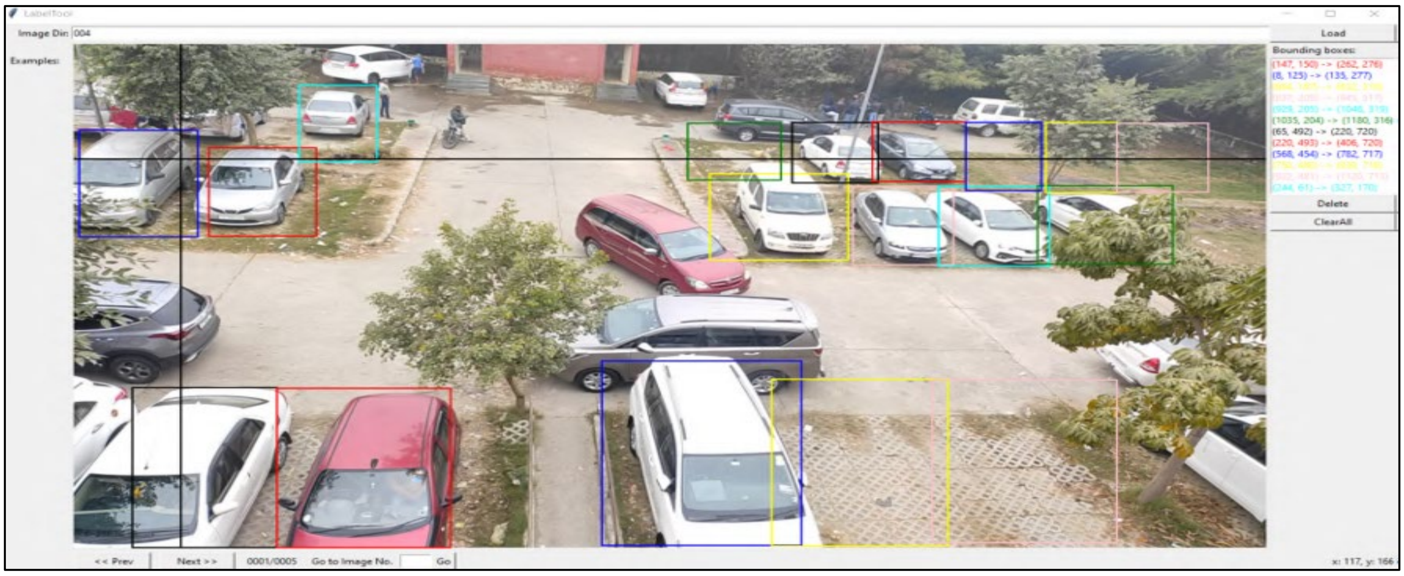
The 'main.py' python file draws the bounding box around objects on the image and stores top-left and bottom-right points in the txt file format. Example of txt file below:

```
2
297 128 367 316
352 116 388 228
```

Fig 23: Annotation data using Yolo Annotation Tool

The convert.py and process.py python files are used to convert the annotations in the specific format used by the YOLO algorithm in the training process.

The process of marking the parking slots is represented as follows-



Bounding boxes:
(147, 150) -> (262, 276)
(8, 125) -> (135, 277)
[illegible]
[illegible]
(929, 205) -> (1046, 319)
(1035, 204) -> (1180, 316)
(65, 492) -> (220, 720)
(220, 493) -> (406, 720)
(568, 454) -> (782, 717)
[illegible]
[illegible]
(244, 61) -> (327, 170)

Fig 24: Use of Yolo Annotation Tool

Here, the first row gives the class id value, and the rest of the rows provide the left-upper and right-bottom points of the corresponding marked parking space coordinates.

- *Mask-RCNN based Parking Occupancy System*

Mask R-CNN provides a wealth of data for each detected object. Most object detection algorithms only return the object's boundary. However, Mask R-CNN provides not only the object's location but also a mask around that object, i.e., object segmentation.

'Common Objects in Context' (MS COCO) is a widely used dataset consisting of images annotated with object masks. By using the above coordinates, the location of each parking space is known. By looking at multiple frames of video in succession, we can easily work out which parking space is occupied or unoccupied. However, for the boxes partially occupied by the car, we need a method to measure how much two objects overlap to predict the occupancy. So, we use a concept called 'Intersection Over Union' or 'IoU'.

$$IoU = \frac{Area\ of\ Intersection}{\text{Area of Union}} \tag{6}$$

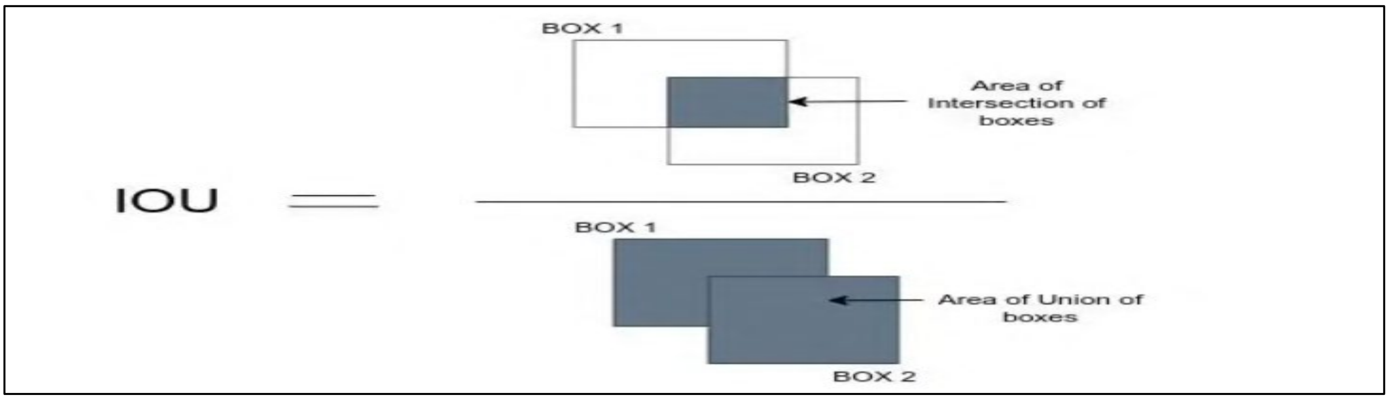


Fig 25: Intersection over Union

For this project, the IoU value is set to be 0.20, that is if the car bounding box occupies more than 1/5 part of the marked parking space, then that space is occupied parking space otherwise unoccupied parking space. If any parking space is continuously unoccupied for more than 100 frames, then the "Space Available" message is written on the frame using OpenCV.

All the frames are passed through the system and stored in an empty folder, and after that, all are merged to get a final video file corresponding to the final output by setting up the required FPS.

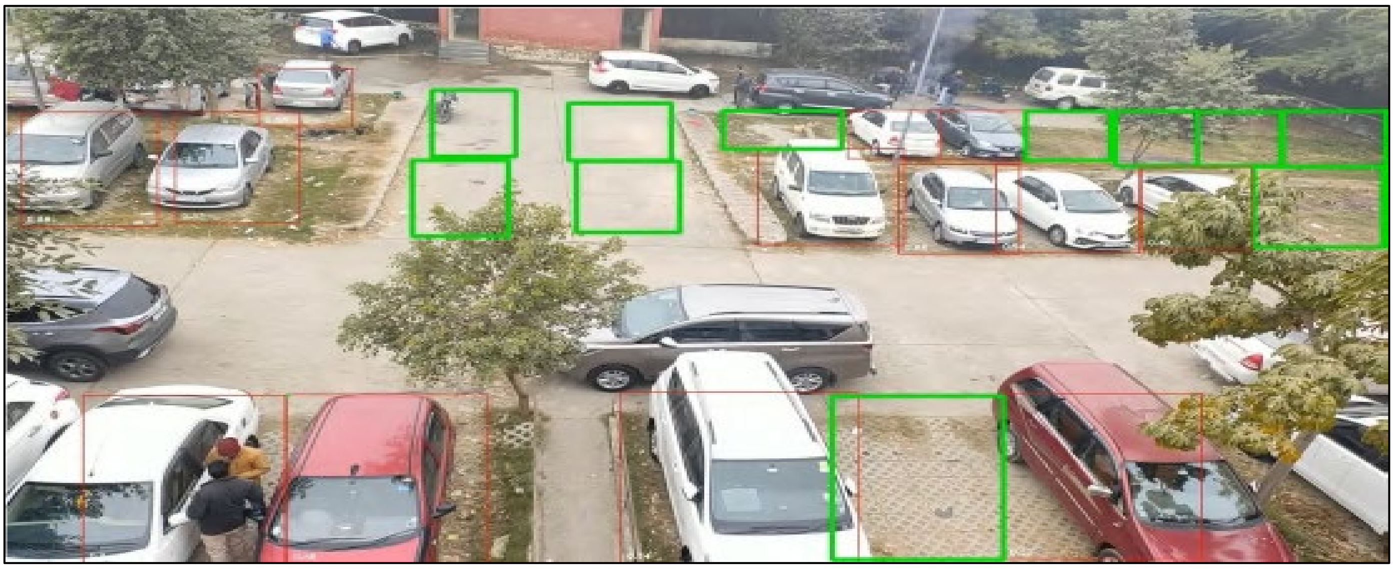
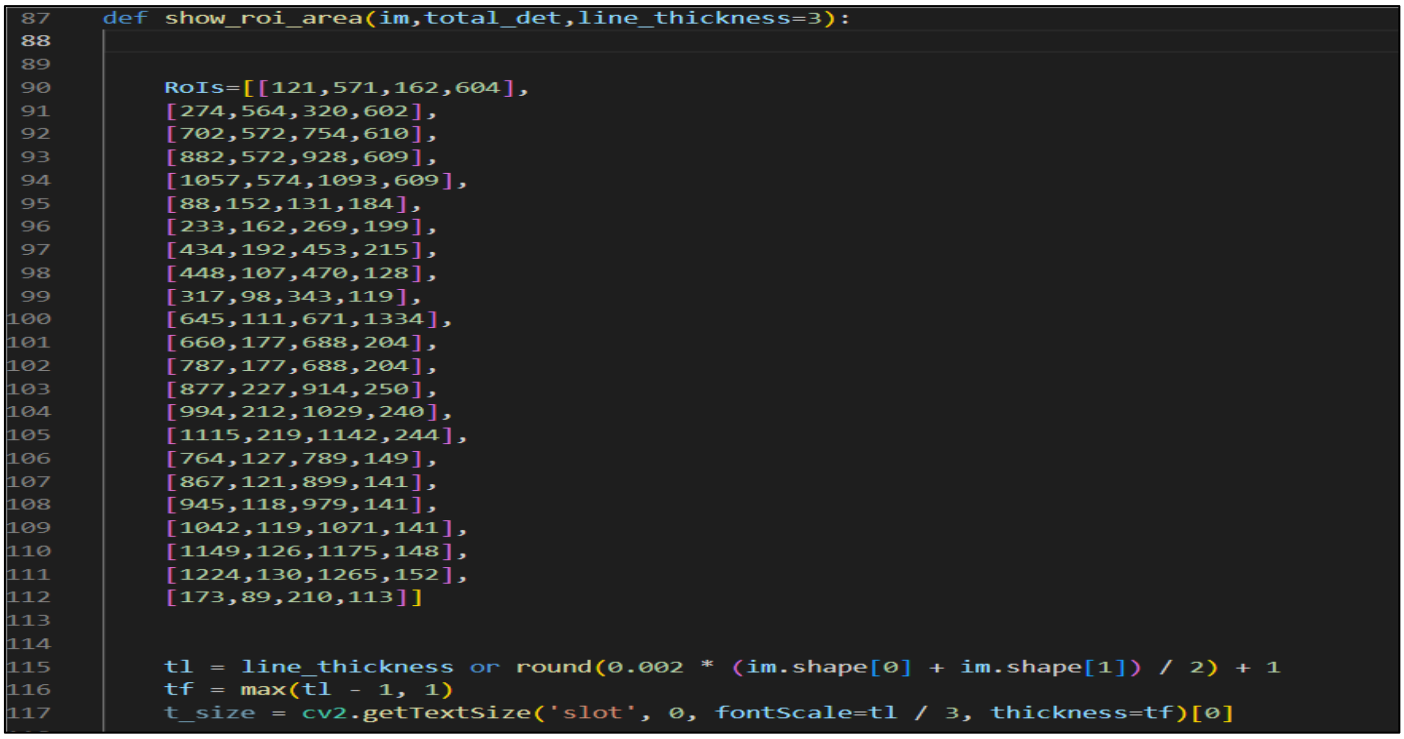


Fig 26: Mask RCNN based Parking Management System

- *YOLOv5 based Parking Occupancy System*

YOLOv5[33] is the recent object detection system developed by "Ultralytics" on the Pytorch framework and was released in June 2020. To locate the occupied and unoccupied parking spaces, YOLOv5 pre-trained model 'yolov5m' is used, which is trained on the "COCO" dataset.The parking spaces marked through the Yolo Annotation Tool are linked with the YOLOv5 algorithm by creating a function in the plots.py file in the Utils folder of YOLOv5 as shown in the following-



```
def show_roi_area(im,total_det,line_thickness=3):


    RoIs=[[121,571,162,604],
    [274,564,320,602],
    [702,572,754,610],
    [882,572,928,609],
    [1057,574,1093,609],
    [88,152,131,184],
    [233,162,269,199],
    [434,192,453,215],
    [448,107,470,128],
    [317,98,343,119],
    [645,111,671,1334],
    [660,177,688,204],
    [787,177,688,204],
    [877,227,914,250],
    [994,212,1029,240],
    [1115,219,1142,244],
    [764,127,789,149],
    [867,121,899,141],
    [945,118,979,141],
    [1042,119,1071,141],
    [1149,126,1175,148],
    [1224,130,1265,152],
    [173,89,210,113]]


    tl = line_thickness or round(0.002 * (im.shape[0] + im.shape[1]) / 2) + 1
    tf = max(tl - 1, 1)
    t_size = cv2.getTextSize('slot', 0, fontScale=tl / 3, thickness=tf)[0]
```

Fig 27: Region of Interests in YOLOv5 Algorithm

The 'detection.py' is also modified as a parking project requirement, using OpenCV to count and show the total count of the vehicles present on each frame and the status of empty and occupied parking spaces available. The code requires running this code specifically for the 'Car' as an object only.

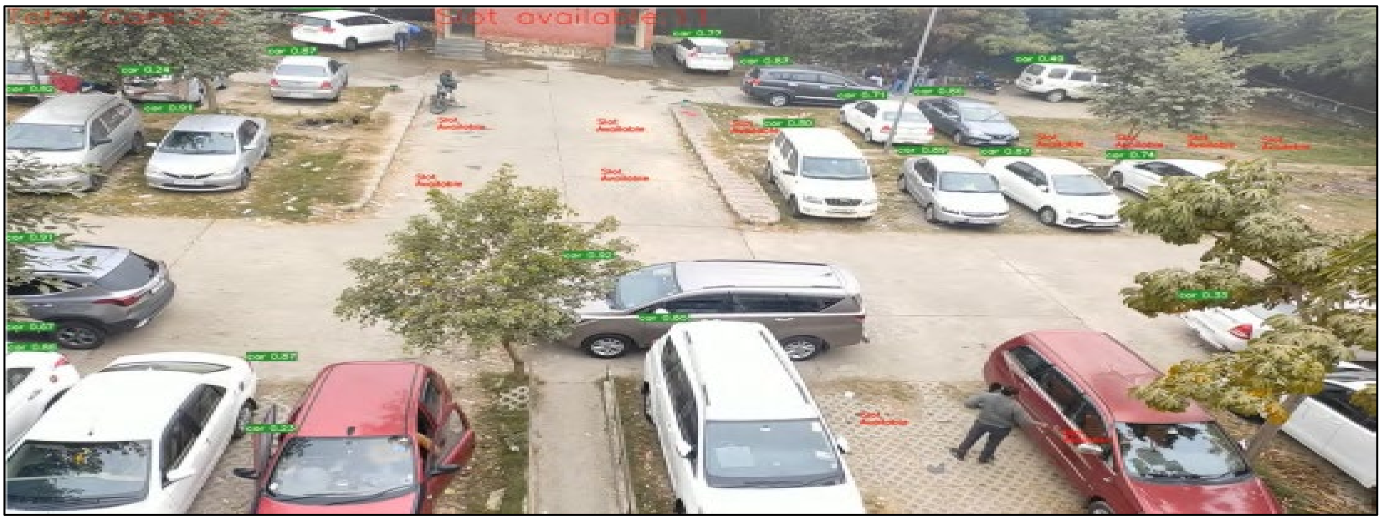


Fig 28: YOLOv5 based Parking Occupancy Detection System

A pre-trained model on the COCO dataset (yolov5m.pt) is used to detect the total number of cars present on each frame. The results are computed by considering various hyperparameters tuning as the confidence threshold (0.25), image size (640), IoU threshold (0.50), and the class of the detecting object. OpenCV is used to count the total number of available cars at any frame as well as the total number of marked empty parking spaces.

Here, all the vehicles present in each frame are counted and mentioned in the left upper corner of the processed image and the total number of marked empty parking spaces is shown in the right upper corner of each processed frame. If the car is not present at any of the marked places, the 'Space Available' message appears at that place.

- *Parking Data Analysis*

Since we have created CSV files containing different parking information, this data is being analyzed to get insights from the data, and day-wise and week-wise analyses are done. This data is also shared with QGIS shapefile layer attributes with their specific layer polygons to represent the insights in geographic representation. Choropleth maps, line charts and area charts are designed to identify hidden patterns and draw conclusions.

✓ *Weekly Area Chart Analysis:*

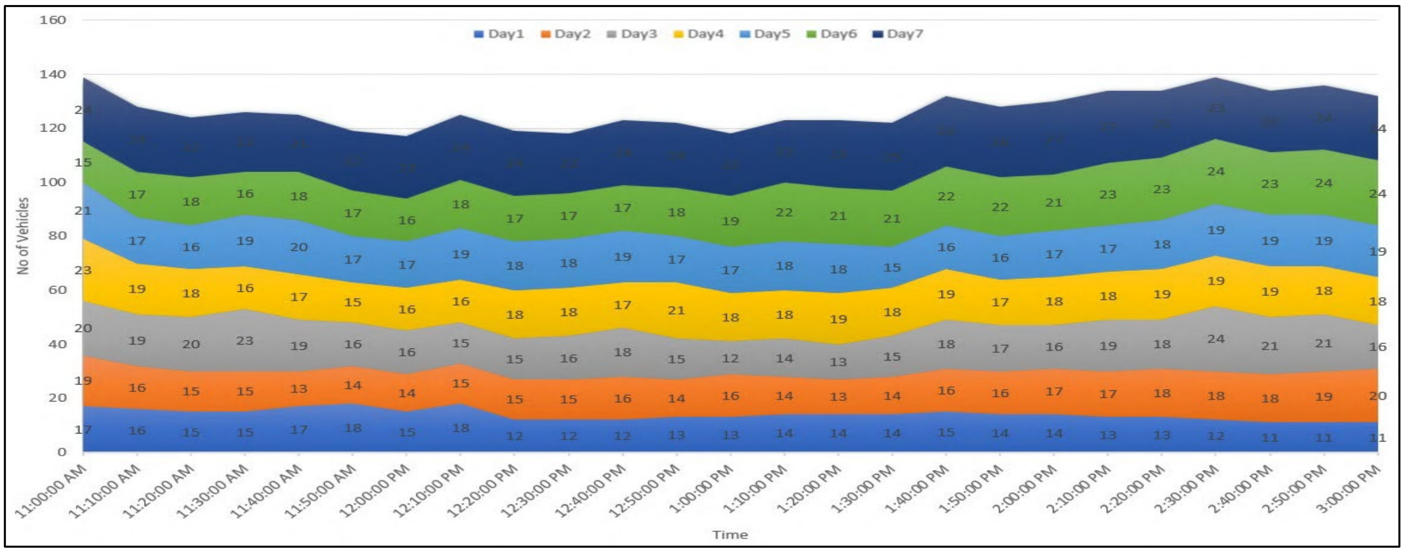


Fig 27: Area Chart Week 1 - Count of Vehicles w.r.t. time

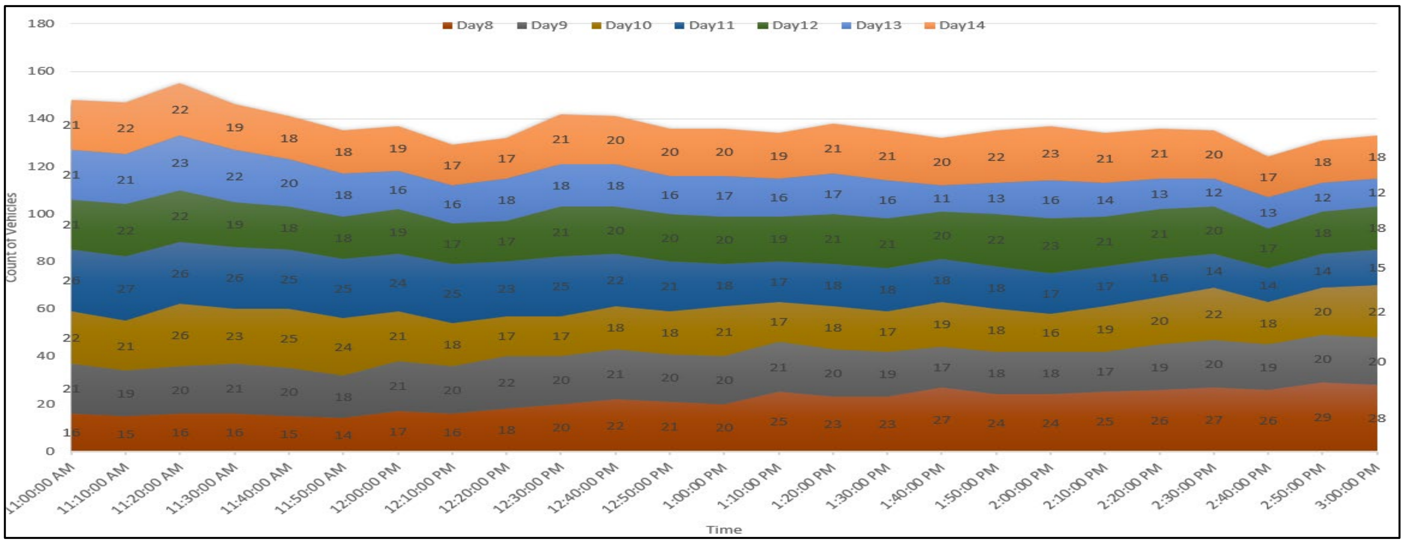


Fig 28: Area Chart Week 2 - Count of Vehicles w.r.t. time

❖ *Insights:*

The line chart and area chart display the number of available cars in the parking lot with respect to time. The following insights can be observed by visualizing the above line and area charts.

- The parking area is sufficiently occupied for the whole 4 hours duration for each day.
- On average, in the morning time i.e. at 11:00 am, there is a greater number of total cars present as compared to any other time.
- In the first week, the average number of cars available in the afternoon was less as compared to the morning time and increased later. In the next week, in the morning time, the average number of available vehicles is more as compared to later.

✓ *Choropleth Map Analysis*

Based on numerical data, choropleth maps use a variety of shading and color schemes. Equal intervals, quantiles, lovely breaks, and natural breaks can all be used to calculate the number of classes. The Equal Interval function divides the attribute value range into equal-sized subranges. For data with well-known fields, the equal interval classification should be used because it highlights how much of an attribute there is in relation to other values.

✓ *Parking Slot Utilization Frequency*

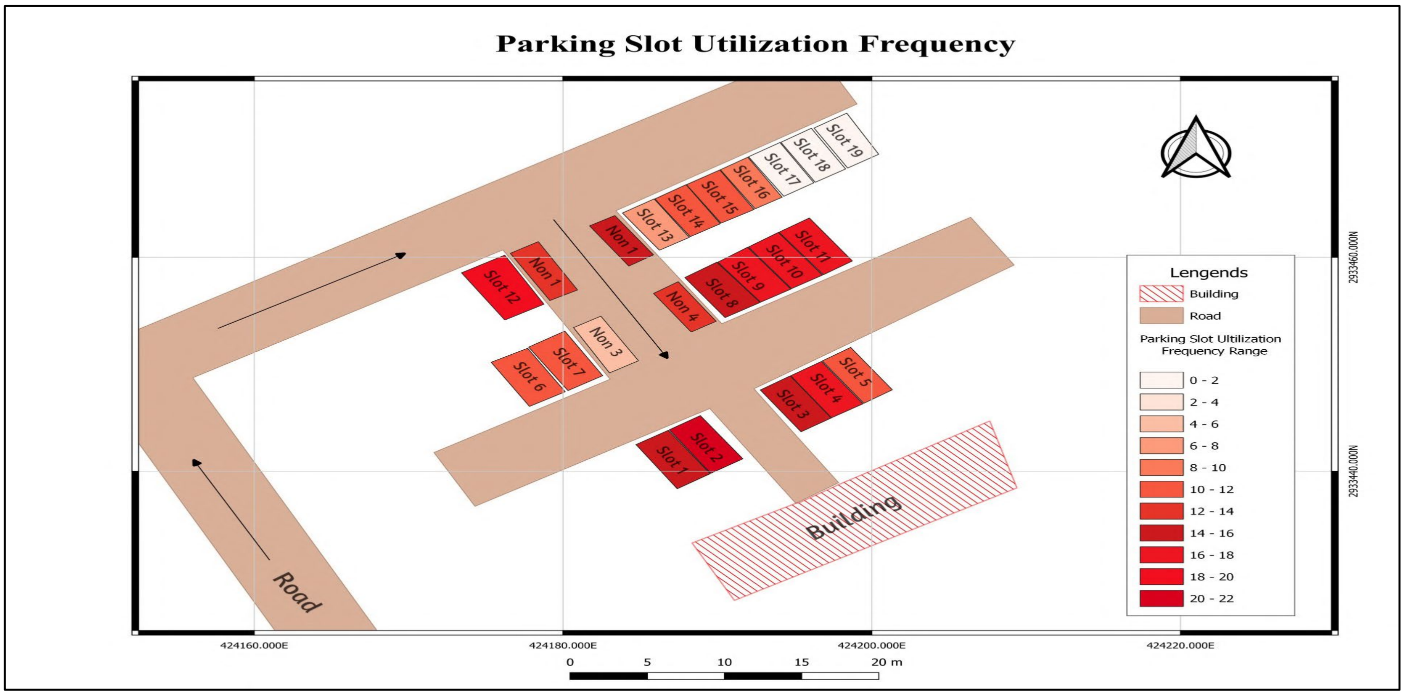


Fig 29: Parking Slot Utilization Frequency Considering 14 Days Dataset

The above choropleth map represents the utilization of pre-defined parking slots by different vehicles over 14 days. The range interval is set to 20 minutes. The color map shows the rate of parking space utilization for each parking space.

❖ *Insights:*

In the above choropleth map, parking spaces are called with the colourmap unit. The shades represent the count of vehicles occupying the respective parking spaces. The following insights are observed from it.

- During the whole 14 days, parking space numbers 17,18 and 19 were rarely used by patrons to park their vehicles as parking slots are at the end of the parking lot and are far from the building. These parking spaces are used mostly only 2 times during the whole 14 days.
- Temporary parking space number 3 is also less utilized as it's much more time consuming to park the vehicle at that location if the Non1 and Non2 parking spaces are occupied earlier.
- Mostly, those parking spaces are used most of the time, which are nearer to the building.
- Due to maintenance work at parking slot number 13 for some days, its use is also less.

✓ *Weekly Average Occupancy of Parking Spaces*

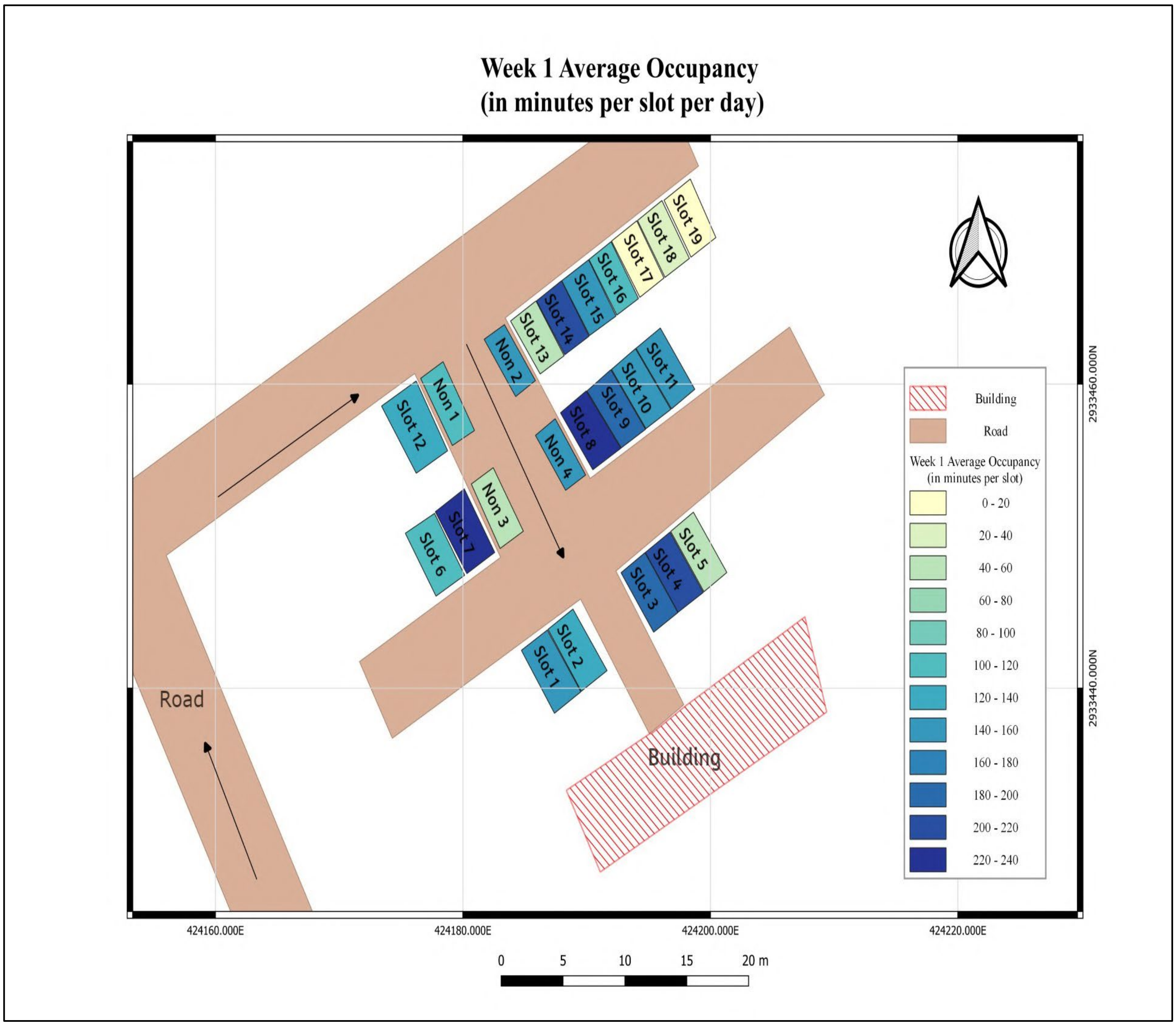


Fig 30: Week 1 Average Occupancy of Parking Spaces

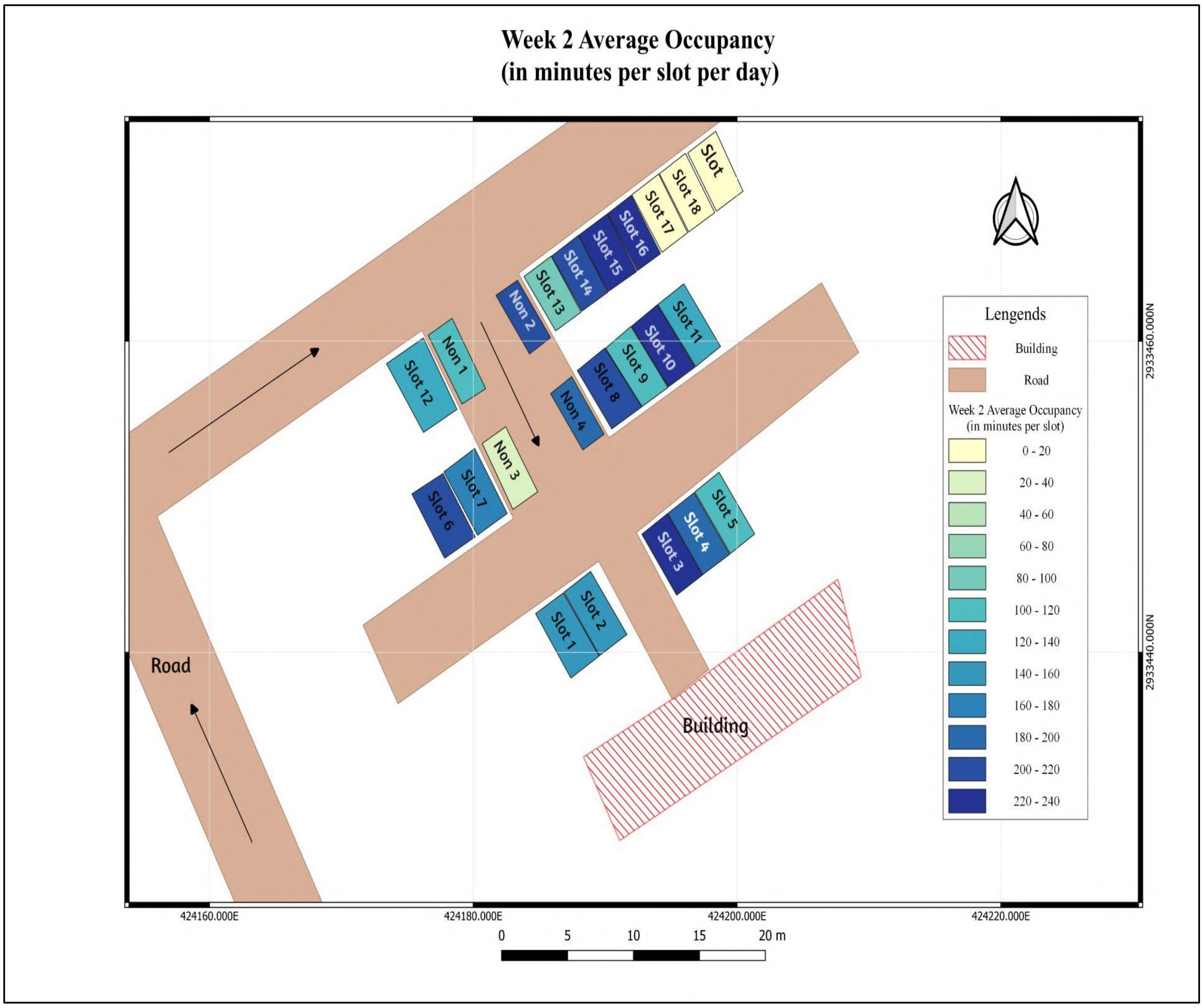


Fig 31: Week 2 Average Occupancy of Parking Spaces

❖ *Insights*

The above choropleth maps represent the average occupancy duration for each of the slots per day by using the color map. The yellow color shows the least value, which progressed towards blue color representing the maximum value.

- In the first week, slot number 17 and 19 are the least occupied as compared to other parking spaces.
- In the next week, slot number 17, 18 and 19 are least occupied as they are at the end of the parking lot.
- Due to maintenance work at slot number 13, its total parking duration is also less in both weeks.
- On average, those parking spaces are more occupied, which are nearer to the connecting road or to the building.

V. SIGNIFICANCE OF THE WORK

This research venture fulfils its objective to develop a functional system that could take the input data captured from a camera in the form of videos and images (frames) and process them to predict the occupied and unoccupied parking spaces within the parking lot. Until this section, the work on this project has met the requirement by utilizing object detection algorithms and OpenCV. The approach proposed in this work takes the input as an image or video, uses the Yolo Annotator Tool to locate the parking spaces, and passes it through an object detection algorithm to get the parking occupancy status. QGIS software is also used to create the shapefile of the parking spaces and merge it with the CSV data generated from the processed frames or videos to get insights from the parking data. This kind of system forms an end-to-end architecture for real-world and real-time applications. Additionally, it establishes a baseline against which future technologies can be evaluated.

VI. LIMITATIONS OF THE WORK

The most significant shortcoming of the work is relying on a single sensor, a camera. Thus, the system is dependent on lighting and atmospheric conditions. Since the data has been collected only during the daytime, the analysis is only related to the daytime situation. The data is collected for a shorter duration of 4 hours for 14 days, which is not significant enough to draw resilient conclusions. There are some novel object detection algorithms that are more efficient but are not used due to their increased computational complexity.

Furthermore, computational complexity analysis and feature visualization, which can help to understand which parts of the network most contribute to the final predictions, are absent from this work. There is also a lack of comparative research into training times as a function of GPU resources and an in-depth analysis of GPU resource utilization.

FUTURE PROSPECT

Since the approach mentioned in this work is in its initial stages toward the final goal, many advancements can be made. Some of which are discussed subsequently.

The network architectures employed in this project consist of classic object detection algorithms, i.e., Mask RCNN and YOLOv5. There are some other nascent object detection algorithms that are much more time-efficient and cost-effective.

A camera is a device that records visual data, usually in the form of images or videos, and is frequently employed for tasks like object detection and classification. Contrarily, a multimodal system incorporates various types of sensors or information sources, such as cameras, 'Light detection and ranging' (LiDAR), radar, or other sensors, to increase the system's accuracy and robustness.

Video observation using an object tracking algorithm may help to obtain individual vehicle trajectories as well as the transition of the parking lot occupancy. This data might be useful to get information related to considering the factors that are important to select to analyze the parking behavior of the patrons in the parking lot.

ACKNOWLEDGMENT

I want to thank my friends Prashant, Kiran, Diksha, Govind, Ibaad and Unnat for their help in various parts of my research and for making my stay on the campus memorable. I would especially want to thank Prashant for helping me to get introduced to all the initial details of the thesis objectives and possible outcomes and methods and Diksha for the support and help during the time of articulation, and for providing useful corrections. Also, I would like to thank my seniors Sujata, Avadh, Rupesh, and Milaa ma'am for their support and time. I cannot describe in words the motivation, fun, and encouragement I have received from these people.

REFERENCES

[1]. B. R. C. Software, "India Automobiles," september 2022. [Online]. Available: https://www.ibef.org/industry/india-automobiles. [Accessed 2022].

[2]. S. A. a. K. C. Shaheen, "Smart parking linked to transit: Lessons learned from field test in san francisco bay area of california," *Transportation Research Record,* vol. 1, no. 2063, pp. 73-80, 2008.

[3]. M. I. L. Y. T. E. N. N. R. Z. Idris, "Car park system: A review of smart parking system and its technology.," *Information Technology Journal,* vol. 8, no. 2, pp. 101-113, 2009.

[4]. jensen, T. S. B. and N. , "Parking space verification: improving robustness using a convolutional neural network," *International Conference on Computer Vision Theory and Applications,* vol. 8, no. 3, pp. 311-318, 2018.

[5]. Anusooya, G. J. J. C. S. and K. , "Rfid based smart car parking system," *International Journal of Applied Engineering Research,* vol. 12, no. 17, pp. 6559-6563, 2017.

[6]. S. A. A. S. S. H. and A. , "A review on smart iot based parking system," *International Conference On Soft Computing And Data Mining,* pp. 264-273, 2020.

[7]. Khanna, A. a. A. and R. , "Iot based smart parking system," *IEEE,* pp. 266-270, 2016.

[8]. Lopez, M. Griffn, T. Ellis, K. Enem and D. , "Parking lot occupancy tracking through image processing," *CATA,* pp. 265-270, 2019.

[9]. Trivedi, J. Devi, M. S. and Dhara, "Canny edge detection based realtime," *Transport/Politechnika slaska,* 2020.

[10]. Blumer, K. Halaseh, H. R. Ahsan, M. U. Dong and Mavridis, "Cost effective single-camera multi-car parking monitoring and vacancy detection towards real world parking statistics and real-time reporting," *International Conference on Neural Information Processing,* pp. 506-515, 2012.

[11]. de Almeida,, P. R. L., Alves, J. H. Parpinelli and Barddal, "A systematic review on computer vision-based parking lot management applied on public datasets," *Expert Systems with Applications,* pp. 116-731, 2022.

[12]. Li, Chen and Ngan, "Simultaneously detecting and counting dense vehicles from drone images," *IEEE Transactions on Industrial Electronics,* vol. 66, no. 12, pp. 9651-9662, 2019.

[13]. Li, X. Chuah, M. C. and Bhattacharya, "Uav assisted smart parking soluton," *international conference on unmanned aircraft systems IEEE,* pp. 1006-1013, 2017.

[14]. De Almeida, P. R. Oliveira, L. S. B. J. A. S. Silva Jr and Koerich, "Pklot{a robust dataset for parking lot classification)," *Expert Systems with Applications,* vol. 42, no. 11, pp. 4937-4949, 2015.

[15]. Amato, G. Carrara, F. Falchi and F. Gennaro, "Deep learning for decentralized parking lot occupancy detection," *Expert Systems with Applications,* vol. 72, pp. 327-334, 2017.

[16]. Amato, Carrara, Falchi, F. Gennaro and Vairo, "Car parking occupancy detection using smart camera networks and deep learning.," *IEEE,* pp. 1212-1217, 2016.

[17]. Nieto, Garcia-Martin, Hauptmann and Martinez, "Automatic vacant parking places management system using multicamera vehicle detection," *IEEE Transactions on Intelligent Transportation Systems,* vol. 20, no. 3, pp. 1069-1080, 2018.

[18]. Dalal and B. Triggs, "Histograms of oriented gradients for human detection.," *IEEE computer society conference on computer vision and pattern recognition,* vol. 1, pp. 886-893, 20005.

[19]. M. A. Suwignyo, Setyawan and B. Yohanes, "Parking space detection using quaternionic local ranking binary pattern.," *IEEE,* pp. 351-355, 2018.

[20]. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," *IEEE conference on computer vision and pattern,* pp. 770-778, 2016.

[21]. J. Yosinski, j. Clune, Y. Bengio and H. Lipson, "How transferable are features in deep neural networks?," *Advances in neural information processing systems,* vol. 27, pp. 324-330, 2014..

[22]. X. Ding and R. Yang, "Vehicle and parking space detection based on improved yolo network model.," *Journal of Physics: Conference Series,* vol. 1325, p. 012084, 2019.

[23]. M. Merzoug, A. Mostefaoui and A. Benyahia, *ACM International Symposium on QoS and Security for Wireless and Mobile Networks,* pp. 37-42, 2019.

[24]. W. Zhao and J. Ma, "End-to-end scene text recognition with character centroid prediction," *eural Information Processing: 24th International Conference,* vol. 3, no. 224, pp. 14-18, 2017.

[25]. Z. Zou, Z. Shi, Y. Guo and J. Ye, "Object detection in 20 years: A survey," *arXiv,* 2019.

[26]. C. Bishop, "Neural networks and their applications," *Review of scientific instruments,* vol. 65, no. 6, pp. 18.3-1832, 1994.

[27]. J. Brownlee, "A gentle introduction to transfer learning for deep learning," *Machine Learning Mastery,* vol. 20, 2017.

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

[29]. K. He, G. Gkioxari, P. Dollar and R. Girshick, "Mask r-cnn," *Proceedings of the IEEE international conference on computer vision,* pp. 2961-2969, 2017.

[30]. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unified real-time object detection.," *Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 779-788, 2016.

[31]. W. Liu, D. Anguelov, D. Erhan and C. Szegedy, "Ssd: Single shot multibox detector," *European conference on computer vision springer,* pp. 21-37, 2016.

[32]. N. Katuk, N. Zakaria and K. Ku-Mahamud,, "Mobile phone sensing using built-in Camera," 2019.

[33]. J. Solawetz, "What is yolov5? a guide for beginners.," 2020".