*A project report on*

# SMART PARK - AI ENHANCED PARKING SYSTEM

*Submitted in partial fulfillment for the award of the degree of*

## Bachelor of Technology in Electronics And Computer Engineering

*by*


**DIVYANSH RAWAL (21BLC1123)**

**AKARSHIT MISRA (21BAI1597)**

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

**SCHOOL OF ELECTRONICS ENGINEERING**


November, 2024

# SMART PARK - AI ENHANCED PARKING SYSTEM

*Submitted in partial fulfillment for the award of the degree of*

## Bachelor of Technology in ELECTRONICS AND COMPUTER ENGINEERING

*by*


**DIVYANSH RAWAL (21BLC1123)**

**AKARSHIT MISRA (21BAI1597)**


Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

**SCHOOL OF ELECTRONICS ENGINEERING**

November, 2024

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

## DECLARATION

I hereby declare that the thesis entitled "SMART PARK - AI ENHANCED PARKING SYSTEM" submitted by DIVYANSH RAWAL (21BLC1123), for the award of the degree of Bachelor of Technology in Electronics And Computer Engineering, Vellore Institute of Technology, Chennai is a record of bonafide work carried out by me under the supervision of Dr. Bharathi Raja S.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Chennai

Date: **Signature of the Candidate**

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

**School of Electronics Engineering**

# CERTIFICATE

This is to certify that the report entitled **“Smart Park – AI ENHANCED DIGITAL PARKING SYSTEM”** is prepared and submitted by **Divyansh Rawal (21BLC1123)** to Vellore Institute of Technology, Chennai, in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Electronics And Computer Engineering** is a bonafide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma and the same is certified.

Signature of the Guide:

Name: Dr./Prof.

Date:

| | |
|---|---|
| Signature of the Examiner | Signature of the Examiner |
| Name: | Name: |
| Date: | Date: |

**Approved by the Head of Department,**
**B.Tech. Electronics And Computer Engineering**

Name: **Dr. Annis Fathima A**

Date:

# ABSTRACT

The "Smart Park - AI Enhanced Parking System" addresses urban parking challenges through an intelligent, automated solution that integrates computer vision and machine learning. Smart Park system utilizes the YOLO v5 model to detect real-time parking spot availability via video analysis, improving parking efficiency and reducing congestion.
YOLO (You Only Look Once) processes entire images in a single pass, making it faster and more efficient than other object detection methods that require multiple image passes or region proposals. This efficiency is crucial in a parking system, where timely updates can minimize wait times and reduce congestion. It uses convolution neural networks to locate and classify objects (in this case, vehicles) in real-time.

The high accuracy of YOLO v5 ensures that free spaces are correctly identified, Additionally a predictive analytics module forecasts parking demand based on historical data, helping users anticipate space availability. The system features a natural language processing (NLP)-driven chatbot for seamless user interaction, offering real-time guidance and booking options. Expected outcomes include high detection accuracy, reliable predictions, enhanced user experience, optimized resource management, and scalability for urban deployment.

# ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to Dr.Bharathi Raja S, Associate Professor, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, for his constant guidance, continual encouragement, understanding; more than all, he taught me patience in my endeavor. My association with him is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Deep Learning.

It is with gratitude that I would like to extend my thanks to the visionary leader Dr. G. Viswanathan our Honorable Chancellor, Mr. Sankar Viswanathan, Dr. Sekar Viswanathan, Dr. G V Selvam Vice Presidents, Dr. Sandhya Pentareddy, Executive Director, Ms. Kadhambari S. Viswanathan, Assistant Vice-President, Dr. V. S. Kanchana Bhaaskaran Vice-Chancellor, Dr. T. Thyagarajan Pro-Vice Chancellor, VIT Chennai and Dr. P. K. Manoharan, Additional Registrar for providing an exceptional working environment and inspiring all of us during the tenure of the course.

Special mention to Dr. Ganesan R, Dean, Dr. Parvathi R, Associate Dean Academics, Dr. Geetha S, Associate Dean Research, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai for spending their valuable time and efforts in sharing their knowledge and for helping us in every aspect.

In jubilant state, I express ingeniously my whole-hearted thanks to Dr. Sweetlin Hemalatha C, Head of the Department, B.Tech. CSE with Specialization in Artificial Intelligence and Machine Learning and the Project Coordinators for their valuable support and encouragement to take up and complete the thesis.

My sincere thanks to all the faculties and staffs at Vellore Institute of Technology, Chennai who helped me acquire the requisite knowledge. I would like to thank my parents for their support. It is indeed a pleasure to thank my friends who encouraged me to take up and complete this task.

Place: Chennai

Date: **Divyansh Rawal**

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

## Results Table

# LIST OF ACRONYMS

1. **YOLO** - You Only Look Once
2. **PIL** - Python Imaging Library
3. **NLP** - Natural Language Processing
4. **AI** - Artificial Intelligence
5. **ML** - Machine Learning
6. **CNN** - Convolutional Neural Network
7. **WSN** - Wireless Sensor Network
8. **SVM** - Support Vector Machine
9. **RNN** - Recurrent Neural Network
10. **LSTM** - Long Short-Term Memory
11. **IoT** - Internet of Things
12. **CVPR** - Conference on Computer Vision and Pattern Recognition
13. **OCR** - Optical Character Recognition
14. **CNRPark** - CNR Institute of Informatics and Telematics (Parking Dataset)
15. **PKLot** - Parking Lot Dataset
16. **CV** - Computer Vision
17. **DRL** - Deep Reinforcement Learning
18. **IIoT** - Industrial Internet of Things
19. **OpenCV** - Open Source Computer Vision Library
20. **NLTK**: Natural Language Toolkit
21. **ARIMA**: Autoregressive Integrated Moving Average (a time series forecasting model)
22. **POS**: Part of Speech
23. **OS**: Operating System
24. **mAP** - Mean Average Precision
25. **MAE** - Mean Absolute Error
26. **RMSE** - Root Mean Square Error

# CHAPTER 1
# INTRODUCTION

## 1.1 PROBLEM STATEMENT

It has been very difficult for urban cities to manage parking facilities with the expansion of vehicles and limited parking facilities. Traditionally, with lesser use of real-time data and much dependence on human supervision, the traditional parking systems introduce inefficiencies such as expanded search times for open parking spots, increased traffic congestion, and wasted fuel. Drivers are unable to find open parking; thus, they get frustrated and waste fuel with unnecessary idling and driving. At the same time, smart and automated solutions are being increasingly demanded to provide real-time information on parking space availability and facilitate a user-friendly experience to its users.

The current parking management systems are largely non-automated, which brings about many inefficiencies and operational challenges. Human involvement in monitoring and managing parking spaces often lacks the precision and speed required in busy urban environments, thereby creating a lack of control over the availability and allocation of parking spots. Moreover, real-time data about available parking spaces adds to the complexity of the matter. Without any live information, the drivers will not know which spaces are free and have to find them all by trial and error, costing so much time spent around driving at random. At the same time, such hopeless searches for parking spots cause frustration to the drivers and add to growing traffic congestion on the roads. The repeated circle-around search for available spots has a negative impact on urban congestion, which further leads to rising emissions and higher fuel consumption levels.

In this context, there clearly exists an urgent need for smart intelligent, automated systems for better management of parking spaces. With advanced sensor technologies, camera systems, and data analytics, a smart parking solution monitors the availability of parking space in real-time and makes available accurate information to drivers so that they may use parking facilities properly. These systems would be automated, therefore minimizing human intervention would decrease errors and enhance efficiencies in the parking process. In this sense, intelligent parking management systems may help cities alleviate the problems of traffic congestion, reduce wastage of time, and create a more streamlined and user-friendly parking experience for everyone.

## 1.2 LITERATURE SURVEY

### 1) Kasera and Acharjee's 2022 article "Parking Slot Occupancy Prediction Using LSTM,"

**Overview**

Publication in Innovations in Systems and Software Engineering: This document explores LSTM networks for the task of predicting the availability of parking spaces. This research demonstrates promise that deep learning models will prove useful in solving dynamic urban parking problems, especially when using temporal information for accurate and adaptive prediction.

**Key Contributions**

Application of LSTM to Predict Time Series: The research demonstrates the ability of LSTM networks to learn and capture long-term dependencies between samples of parking data. This feature is important in predictive parking systems since understanding how past patterns of occupancy influence future availability is essential.

**Improved Prediction Accuracy:** The authors compare the LSTM with the traditional statistical and machine learning models in the context of predicting parking slot occupancy, showing its superiority. It also supports robustness against noisy patterns and irregular spikes in demand in real-world application scenarios.

**Potential for Real-Time Implementation:** The paper establishes the potential of the model in actual real-time, on-ground parking management systems. With the precision in predicting parking assignments, drivers can efficiently plan their parking decisions hence saving time in searching for available parking spaces and dissipating congestion.

**Applications and Implications**

An LSTM-based predictive model can be added to existing parking management systems or mobile applications to provide real-time guidance to drivers. Its potential for adaptation can also be used in the context of dynamic pricing models, in which prices are updated according to predicted demand. Additionally, insights from this model can be used in designing more efficient parking infrastructure by urban planners.

**Challenges and Future Work**

The authors do recognize challenges such as the availability and quality of parking data, especially in cities where digital parking systems are not widely spread. They also mention that using more additional sources of data, such as the weather forecast or traffic flow, would further improve prediction accuracy. An area for further research relates to optimizing the computational efficiency of large-scale LSTMs.

**2) Kazi, Nuzhat, Nashrah, and Rameeza's 2018 paper, "Smart Parking System to Reduce Traffic Congestion,"**

Presented at the International Conference on Smart City and Emerging Technologies (ICSCET), deals with designing and implementing a smart parking system to alleviate traffic congestion in urban areas. The paper underlines the necessity of using technology to meet the boom of automobile parking in highly busy cities.

**Overview of the Study**

The proposed research includes an intelligent parking system that makes use of IoT and sensors to offer real-time information about available parking spaces. The system aims to minimize the time in searching for parking, which is the major source of traffic congestion in cities. The proposed solution not only helps enhance the efficiency and convenience to drivers but also benefits traffic flow as a whole by automating the process of detecting and allocating parking spaces.

**Key Contributions**

**IoT-Driven Real-Time Detection:** All of these monitor parking space with the help of ultrasonic and infrared sensors from IoT devices. The system senses vehicle presence and transmits the data to a central system for updating availability in real time.

**User-Friendly Interfaces:** This system is inclusive of a proposed mobile application that will allow users to check parking availability, reserve parking spaces, and further direct them to the nearest available parking location. This diminishes the need to manually search for a parking spot, thereby saving time and reducing vehicle emissions caused by idling and circling around.

**Traffic Congestion Reduction:** The system prevents unnecessary vehicle movement by giving precise, real-time parking information to drivers, contributing a lot to urban congestion. As a result, traffic is brought under more control, and the environment is not polluted to that extent.

**Challenges and Future Scope**

The paper notes scalability and cost concerns when the IoT-based parking solution needs to be implemented in the big cities. The installation and maintenance cost of the infrastructure needed to deploy sensors everywhere could turn out to be an obstacle, especially in developing regions. Future work may seem to reduce costs by technological innovations, and alternative detection methods like computer vision or AI can be explored.

3) **Caicedo, Blazquez, and Miranda's 2012 paper, "Prediction of Parking Space Availability in Real Time,"** published in Expert Systems with Applications, is a foundational study in the domain of predictive parking management. The paper explores the design and implementation of a real-time prediction model to estimate parking space availability, offering practical solutions to urban traffic congestion caused by inefficient parking searches.

**Overview**

The authors introduce a model that predicts parking space availability by combining real-time inputs with historical parking data. This advanced statistical and computational technique measures and computes dynamic forecasts in order to predict parking demand, ultimately serving both drivers and urban planners. The presented approach is relevant particularly for high-density urban environments; timely information about available parking spaces can efficiently reduce congestion as well as help curtail the related environmental impact of vehicle emissions due to protracted searches.

**Key Contributions**

**Real-Time Predictive Framework.** In this paper, the authors introduce a strong framework that will process real-time data pertaining to entry and exit records in parking lots for the prediction of parking availability. The model continuously adapts and responds appropriately to the demand and is highly accurate even in peak periods.

Integration of Historical and Real-Time Data Combining historical occupancy data with current information maximizes the model's ability to predict availability. A hybrid approach enhances adaptability to changing conditions, including peak hours and fluctuations in seasonality.

**Application to Traffic Management:** The model enables a predictive service, so predictions not only for particular drivers but also for whole cities are feasible. Parking demand can be used by cities to apply dynamic pricing or rerouting measures.

**Limitations and Future Work**

The authors discuss the model's shortcomings and limitations, such as the availability and variability of data. For example, unexpected events or rapid changes in demand can affect the accuracy of predictions. Suggests the use of other real-time sources such as traffic sensors and weather forecasts to improve model robustness.

**4) Provoost et al. in their paper "Predicting parking occupancy via machine learning in the web of things,"** published in Internet of Things in 2020, is an important work within the domain of the predictive parking management research area. The paper performs research on how ML methods can be used to predict parking occupancy by taking advantage of the WoT ecosystem, which aims to improve urban mobility and reduce the frustration from inefficient parking searches.

**Overview**

This method produces a model that combines real-time data from IoT-enabled parking sensors with historical data in order to predict parking occupancy. Advanced ML algorithms built into this predictive model work dynamically to predict availability, targeting both the driver and the city planner. The approach comes at the top as quite suitable for smart city initiatives, where connected devices play a critical role in managing urban infrastructure.

**Key Contributions**

**Real-Time Parking Prediction Framework**

It introduced a framework that used IoT sensors to collect data from parking lots and applied ML models to predict occupancy. Designed to be dynamic, it accounts for the changing situations and would ensure accurate forecasts during high variability in demand.

**Combination of Historical and Real-Time Data**

The paper combines historical parking data trends with real-time inputs like time of day, the weather, and occupancy rates to increase the accuracy of prediction. This hybrid approach increases the responsiveness of the system to changes within the context, such as peak hours and seasonal variations.

**Web of Things (WoT) Integration**

The framework thus embeds the predictive system within the WoT, allowing communication between connected devices without interruption; otherwise, the system can run real-time updates and user-friendly interfaces for drivers through mobile apps or navigation systems.

**Limitation and Future Work**

Authors mention that the performance of the model would be very much based on the availability and data quality. Rapid changes in demand or other unforeseen events like special events or accidents could also be an issue that can affect the prediction accuracy. Further, they mention incorporating additional real-time data sources on traffic and weather sensors for increasing robustness, and scaling the solution to dynamic street parking scenarios as potential areas of future research.

## 1.3 PROJECT OVERVIEW

Increased growth in urbanization and car ownership are naturally demanding effective parking management solutions. Current traditional parking systems rely on manual supervision and do not give real-time information, which is partially the reason for inefficiencies like longer search times for parking, increased fuel consumption, and traffic congestion. The Smart Park system answers all these inefficiencies by offering an AI-driven solution to the modern parking management solution, improving parking resources, increasing the user experience, and reducing traffic-related pollution. This system employs some of the latest technologies including computer vision for real-time parking detection, predictive analytics through machine learning, and natural language processing for user interaction.

Urban areas have become bedlam quite recently, especially when it comes to parking, with so many cars and scarce parkings. Traditional methods of parking management rely on human evaluation, guesswork, and labor, failing to meet the standards of modern cities. There is an undesirable consumption of fuel and detrimental environmental effects due to time wasted in searching for available spaces. The lack of sufficient real-time information on availability also causes frustration and contributes to traffic congestion, which further lowers the overall productivity. This project, called Smart Park, was designed to develop an automated and streamlined process for parking through the advanced technologies of parking spot detection, prediction, and user interaction.Smart Park is designed as a modular system comprising three core components: the Computer Vision Module, the Predictive Analytics Module, and the User Interaction Module. Together, these components enable real-time parking detection, accurate availability forecasting, and seamless interaction with users.

**1) Computer Vision Module:**

**Technology:** YOLO v5 (You Only Look Once) Model, Python Imaging Library (PIL)

**Function:** The Computer Vision Module employs the deep functionalities of YOLO v5, thereby enabling users to book in advance; hence drivers do not end up waiting uncontrollably for a spot. NLP libraries such as spaCy and NLTK enable the chatbot to understand and answer queries accordingly with respect to the context.

**Benefits:** It offers an instant communication link between the user and the system. The User Interaction Module brings about a more positive user experience. Easy access to real-time information is enabled through the chatbot while guiding the driver on available spots, thus offering a smooth friction-free parking experience.employs a learning model with high speed and accuracy for detection of available real-time parking spots. The architecture of YOLO v5 is such that it processes images in a single pass. It makes the model very efficient for such real-time applications like parking detection. Once embedded with feeds from live videos of parking areas, the model is capable of distinguishing whether parking spots are occupied or vacant by identifying and categorizing vehicles. PIL supports image pre-processing tasks, including resizing, format conversion for accurate analysis of the YOLO model.

**Advantages:** This module provides a fast and accurate indication of parking space occupancy in the present, thereby offering instant feedback from the system to its users. The module helps in lesser time consumption by drivers in parking, improves traffic flow, and reduces jamming in parking areas.

**2) Predictive Analytics Module:**

**Technology:** Python, Supervised Learning Models, Data Normalization Techniques

**Function:** This module uses the trained machine learning model to predict parking availability. Inputs were made on a common scale on data normalization, thereby enhancing the efficiency of the model. The system gauged occupancy trends and patterns of such areas of scarcity or abundance to predict time and place when and where there is likely to be free parking available. The drivers can prepare in advance for their parking arrangement with this, thus making less unproductive circulation of vehicles within the parking lot.

**Benefits:** First and foremost, this module benefits the users by assisting the driver in predicting availability, and also helps the parking lot operator to allocate space. It empowers efficient use of space and can be scaled for a wide range of urban settings with dependable forecasting.

**3) User Interaction Module:**

**Technology:** Natural Language Processing Libraries (spaCy, NLTK), Integrating the Chatbot

**Function:** The User Interaction Module has NLP abilities that would facilitate a chat-bot interface to which the drivers talk or correspond by voice or text; this would be embedded on infotainment systems or mobile devices and, in real time, update them on parking availability leading them to an open space.

## 1.4 OBJECTIVE

The aim of the SmartPark AI-Enhanced Parking System is to meet the increasing demand for efficient parking management within cities through advanced technologies such as computer vision, machine learning, and natural language processing. The system of SmartPark will reduce the amount of time a car will spend finding an available parking space, provide reduced congestion in parking, and overall improve the use of parking space.

The implementation of the YOLO v5 model will ensure that users are correctly informed of the availability of parking within the facilities, thus, minimizing frustrations and inefficiencies surrounding the use of manual parking systems. The predictive analytics module in SmartPark shall forecast future demands for parking, which means users can plan their parking ahead of time and also achieve less unnecessary circulation in automobiles in parking lots.

Another important objective of SmartPark is to provide the users with a seamless interaction platform through an integrated chatbot. The natural language processing-enabled chatbot shall be meant to answer user queries related to parking availability, navigation, and reservations using a friendly, intuitive interface, which shall be accessiblethrough infotainment systems in the vehicle or on mobile devices. Others target dimensions related to scalability, responsiveness to changing infrastructure needs, and integration with more general smart city infrastructures.

Therefore, SmartPark will seek to build a system that not only minimizes parking resources but also finds a harmonious place in other smart city systems. In that respect, SmartPark seeks to be part of an efficient, integrated urban ecosystem that maximizes traffic flow, resource use, and environmental sustainability.

## 1.5 PROJECT OUTCOME

Improved parking efficiency, user satisfaction, and urban traffic management are expected with the SmartPark project. Significant results to be anticipated from it are:

1. **Improved Detection Accuracy:** The proposed system will achieve efficiency in classifying available parking spots for the facilitation of higher accuracy in spotting, eliminating misclassification conditions, and proper availability information, making use of YOLO v5's advanced image processing system.
2. **Accurate Predictive Insights:** With the help of machine learning-based forecasting, the drivers will have proper predictions about parking availability to plan better and save time.
3. **Enhanced User Experience:** The driven interaction module that combines a chatbot offers users a friendlier interface to enhance driver convenience, providing real-time guidance and booking options for a seamless parking experience.
4. **Optimized use of resources:** SmartPark minimizes traffic congestion in parking lots, provides smooth road traffic flow, encourages environmental sustainability through decreased idling emissions.
5. **Scalability and Flexibility:** the modular nature of SmartPark makes it easy to deploy across various city contexts and can possibly be merged into broader smart city platforms.
6. **Real-Time Analytics for Urban Planning**: The data generated by SmartPark can serve as a valuable resource for urban planners. Real-time insights into parking trends and usage patterns can inform decisions on infrastructure development and traffic management.
7. **Support for Autonomous Vehicles**: The system can be adapted to accommodate autonomous vehicles by providing precise navigation and reservation options tailored to their requirements. This makes SmartPark future-ready as autonomous technologies become more mainstream.

## 1.6 CHALLENGES

One of the toughest challenges when building SmartPark is getting real-time, or nearly real-time, identification of available parking. The model heavily depends on the YOLO v5 model, a high-performance, deep-learning architecture for object detection in video feeds. However, implementing YOLO v5 demands tremendous computational power, especially if continuous, high-resolution video feeds are being analyzed. Another model is the Predictive Analytics Module that predicts parking availability, which comes with a series of its own challenges. Predicting parking demand would require large datasets of historical usage patterns over time. Such data are often tough to be readily available and of sufficient and good quality, especially across different urban locations due to limited historical records or privacy concerns related to tracking vehicle occupancy data.

Parking lots and garages present diverse physical environments with their own unique characteristics, adding another layer of difficulty for SmartPark's real-time detection and forecasting capabilities. For example, daylight can be compared with artificial lighting at night or even changes in weather can be challenging for video feeds which can result in lesser accuracy consistency with YOLO v5.

## 1.7 SCOPE OF THE PROJECT

The scope of the SmartPark project is the bringing up of a highly integrated, AI-orientated parking management system that deals with the needs of urban drivers and operators managing parking facilities. Advanced computer vision and machine learning techniques are expected to provide real-time detection of available parking spots and predictions of future availability. The system's core components—computer vision, predictive analytics, and user interaction modules—enable accurate, live updates on parking occupancy and a chatbot-driven interface for user engagement. Designed for scalability, SmartPark is intended for deployment in diverse urban environments, from multi-level parking garages to outdoor lots, ensuring that its functionality can be adapted to various infrastructure and traffic conditions. This flexibility enables SmartPark to servea wide array of facilities, bringing value to cities, commercial parking operators, and individual drivers.

This will also have the capability to integrate capabilities with other smart city systems and IoT networks to form a connected urban ecosystem. Architectural features include the integration of public transportation networks, environmental sensors, and digital payment systems to support seamless data sharing and interoperability in Smart Park. It is also meant to contribute to broad citywide goals, like reducing traffic congestion, lowering emissions, and improving overall mobility around the city. It's meant to look ahead into the future with advancements, so improvements such as advanced AI models, increased sensor integration, and data-driven enhancements are possible upgrades in the future. In a nutshell, the scope of Smart Park extends beyond basic parking management to being a fundamental element within a more elaborate, integrated smart city framework to improve urban quality of life.

# CHAPTER 2

# BACKGROUND

## 2.1 INTRODUCTION

Demand for parking continues to be on the upside compared to supply with the increasing expansion of urban areas and vehicle ownership. Drivers worldwide spend considerable amounts of time and fuel resources in search for vacant parking spaces, and this contributes not only to individual inconvenience but to the broader urban challenges such as traffic congestion and air pollution. Because it is these traditional parking management systems that are generally responsible for all these inefficiencies. These systems rely heavily on manual supervision and do not provide real-time data capabilities, leaving drivers without crucial information timely for making decisions about parking. Therefore, the vehicles go round repeatedly, worsening traffic bottlenecks and straining unnecessary environmental pressure. The need for smart, automated parking solutions has never been more obvious.

To these pressing issues, the Smart-Park AI-Enhanced Parking System was developed as an innovative response to urban parking challenges. This system utilizes the state-of-the-art advancement in artificial intelligence, computer vision, and machine learning to transform traditional parking management into a streamlined, data-driven process.

One of the standout features of Smart-Park is its real-time management of parking. It accomplishes this by using live video feeds with computer vision technology to fairly and accurately track the occupancy status of parking areas. As a result, a Smart-Park system could distinguish vacant from occupied spots and provide drivers with up-to-date, precise information.

Further, machine learning algorithms also increase the predictability power of the system. Using historical data, such models can predict future available trends. For example, they could predict the likely peak demand times in specific regions or suggest alternative parking locales based on user behavior patterns.

This predictive functionality enables drivers to plan their parking in advance, reducing unnecessary driving and associated frustrations.Smart-Park takes user convenience a step further with its intuitive chatbot interface.

This AI-powered chatbot interacts with people through a mobile application, providing real-time updates on parking availability and directions to nearby spots and even gives the option to book in advance before arriving at the place. Imagine pulling up in downtown somewhere and opening an app, asking the chatbot what's available, and getting turn-by-turn directions to the nearest vacant parking spot within seconds. This level of automation not only saves one's time but also enhances the whole user experience.

The system's scalability makes it adaptable in many types of parking facilities, from high-demand urban garages to suburban lots and event-specific parking areas. By offering a versatile solution, Smart-Park caters for diverse urban settings, therefore ensuring effectiveness in different environments. In addition to improving personal parking, Smart-Park significantly impacts broader urban sustainability goals.

The system's direct impact on reducing ineffective driving considerably reduces traffic congestion, meaning that traffic flows much more freely in the main city centers. Reduced congestion also means reduced vehicles emissions, an increase in cleaner air, and a more salubrious urban environment.

Environmental advantages include less noise pollution since fewer cars are circulating to find parking areas. Implementing Smart-Park perfectly meshes with the vision aimed at pushing urban spaces to become more "green" and livable. On an economic perspective, municipalities and private parking operators will surely benefit from this. This system maximizes revenue potential through smarter optimization of the given space in parking lots, while simultaneously reducing operational costs by streamlining them through automation. Besides, the data that are collected by the system will be useful for future urban planning. For example, resource allocation decisions or design of new parking facilities, etc .

Smart-Park is paving the way toward intelligent and smarter city infrastructure. Not only does Smart-Park address immediate parking needs, but it also helps shape the cities of the future. As more cities come aboard in utilizing intelligent parking solutions, the cumulative effects are more pronounced in the realm of urban mobility and sustainability.

Drivers experience less frustration, there is a reduction in emissions, and traffic congestion is less of an issue-all of which do contribute to a higher quality of life for the urban dwelling population.

Smart-Park shows, then, technology's possible role in solving real-life problems and making city living more comfortable. Parking management reinvented through the synthesis of AI, machine learning, and computer vision is what Smart-Park does. This innovative parking system breaks all the present standards of efficiency and convenience with its feature of real-time monitoring and predictions.

Ultimately, it will be more than a parking solution. It will be an important step forward toward a connected, sustainable, and intelligent urban ecosystem. Solutions such as Smart-Park will prove crucial in ensuring that future development in cities is compatible with the needs of both human life and natural environment.

## 2.2 EXISTING WORK

**Xiang, et al., Geng, and Cassandras. "AI-driven Parking Systems for Real-time Detection and Predictive Analytics in Urban Management."**

Recent developments in AI-driven parking systems have now surfaced as a promising solution to tackle the challenges that have plagued urban parking management. Several research studies and projects have examined various methods and technologies intended to optimize the efficiency of parking. Notable within this are methods using computer vision to identify real-time vacant parking spots. For instance, in some systems, deep learning models like CNNs are used to analyze video feeds coming from cameras mounted in car parks and monitor the frames in order to track cars and identify whether the spaces are occupied or not. For instance, Xiang et al. illustrated the application of CNN-based models in detecting parking as rapidly and accurately as possible for real-time detection, saving much time for searching drivers for empty spaces. Another important AI-based development in smart parking is predictive analytics, where the model is conditioned on historical parking data to predict future occupancy.

Typically, these models rely on supervised learning algorithms that track patterns in the data, such as peak hours or seasonal trends. Geng and Cassandras looked into how one could apply predictive algorithms to parking and reservation systems, with users advancing their planning according to the forecasted availability.

Predictive systems provide the users and managers of car parks with information on peak and off-peak demand periods, which result in decreased congestion and optimal space utilization with effective resource management. Being data-informed, dynamic predictive models, therefore, are more sophisticated than the typical static approaches because they respond to changes in conditions. Other than these, AI-based parking systems have also drawn attention for its integration of IoT devices and sensors that complement the visuals from cameras and can provide a more holistic account of what is happening in terms of parking.

For example, in certain smart parking systems, ground sensors are installed underneath each parking place to determine if a vehicle exists and then wirelessly report back to a central system, which actually dynamically updates real-time availability maps for the drivers. The authors have found that WSNs in open parking are of utility as low-power sensors for data accuracy and availability, especially in open outdoor environments where lighting and weather may degrade camera-based detection systems.

These systems exploit IoT technology by collecting granular data about the occupancy, thus improving model variants of machine learning for predictive analytics and accuracy in real-time parking detection. One of the industrial players, Parking Telecom, has already introducedintegrated systems that align AI-based parking solutions with digital payment options in one interface. Users can reserve a parking spot and even pay in the very same interface. This single phase makes it really simple and supports its extensive usage in private parking lots and public places.

In general, the existing works on AI-enhanced parking systems illustrate great strides in this field and diverse approaches to core challenges that face parking management.The integration of computer vision, machine learning, IoT devices, and user-centric design in these systems forms the basis on which even more efficiency, accuracy, and experience in urban parking will then be improved. Based on such an advancement, Smart-Park has integrated state-of-the-art technology into designing a holistic parking solution that meets the modern demand of an urban setting.

## 2.3 RELATED WORK

**Predictive Analytics for Parking Demand**

Besides real-time detection, predictive analytics becomes an integral part of AI-based parking systems. Using historical occupancy data as input, machine learning models can predict future demand and availability for them to make proper choices and park in advance. Most pattern identification algorithms used in the area of parking demand are linear regression, decision trees, and SVM. Geng and Cassandras (2013) then created a predictive model that analyzes historical data to predict the occupancy of parking lots and directly helps in the allocation of more-efficient space, plus popularizing reservation systems. This work focuses on peak hour and seasonality forecasting, particularly in high-demand areas, such as city centers and close to public transportation facilities.

However, more recent efforts on advanced machine learning approaches, like ensemble methods and recurrent neural networks (RNNs), have the capability to extract time-series dependencies underlying the parking data. In the purpose required, RNNs, especially long short-term memory (LSTM) networks, are well-fitted to learn patterns from sequential data and find long-term dependencies.

Such models of prediction find suitable applications in smart cities, thus enabling parking operators to predict peak periods and accordingly manage them, thus reducing congestion and better user satisfaction can be assured. However, problems persist, especially related to data quality and how variability affects things such as weather, local events, and holidays, which all affect parking demand.

**Machine Learning Approaches**

Traditional predictive models for parking demand often utilize techniques such as linear regression, decision trees, and support vector machines (SVMs) to identify patterns in parking occupancy.

For example, Geng and Cassandras (2013) developed a predictive model that analyzes historical parking data to forecast lot occupancy. Their work is particularly notable for its emphasis on peak hour and seasonality forecasting in high-demand areas like city centers and transit hubs, contributing to better space allocation and the adoption of reservation systems.

More recent advancements have incorporated ensemble methods and recurrent neural networks (RNNs) to handle the complexities of time-series parking data. Among RNNs, long short-term memory (LSTM) networks are especially suited to capturing sequential patterns and long-term dependencies in parking data. These models offer improved performance by addressing temporal correlations, making them well-suited for applications in urban environments where parking trends exhibit cyclical or irregular fluctuations.

### Sensor Integration with AI

However powerful computer vision is, it can be complemented by IoT-based sensors for better accuracy and reliability, especially in challenging environments where visibility may be obstructed. Other types of sensors, which include magnetic, ultrasonic, and infrared are incorporated into AI-advanced parking systems to recognize vehicle presence. For instance, ground sensors embedded in every parking lot can directly indicate whether a space is occupied or vacant. Vera-Gómez et al. (2016) investigated WSNs for parking lots, showing how lowpower, lowcost sensors can enhance occupancy detection systems, especially outdoors, where lighting and weather condition could affect camera-based detection.

Additionally, the IoT sensors provide more resolution, hence providing real-time updates on the status of occupancy for feeding into machine learning models. The information gathered from sensors and cameras combines to form a stronger system that can produce better accuracy with few false positives. There are systems that even adopt a hybrid approach wherein it utilizes computer vision as the main detection but falls back on sensors in cases of poor conditions. For example, a smart parking system, which integrates camera feeds with ultrasonic sensors' data can remain accurate regardless of the camera's vision, even when partial vision is covered by other cars, rain, or fog.

### YOLO

YOLO was proposed by Joseph Redmon and others in 2016 (Redmond et at., 2016). It is a real-time object detection system based on Convolutional Neural Network. Joseph Redmon and Ali Farhadi published YOLO v2 in the Conference on Computer Vision and Pattern Recognition (CVPR) during 2017, expanding the accuracy and speed of the algorithm (Redmond & Farhadi, 2017). As recent as this April, Joseph Redmon and Ali Farhadi published the latest YOLO v3 that further improved the performance in object detection (Redmond & Farhadi, 2018). To track the inside of a parking lot by vehicles, a YOLO v3 is used in this paper. Vehicle Number Recognition

Vehicle number recognition technology has been widely used already, and in general, deep learning techniques such as Faster R-CNN or YOLO are used (Yu et al., 2020). In this paper, since license plate recognition was not the topic but rather depicting a scene of recognition of vehicle numbers, rather than using deep learning technology, a simple Optical Character Recognition (OCR) method for extracting characters from images was used (OpenCV, n.d.).

Hoon Lee and his colleagues conducted a systematic review of the application of AI and CV technologies in the allocation of parking space, especially in seaports. The research underlines the advantages that the AI and CV produce toward efficient and sustainable parking ecosystems. It underlines the contribution that these technologies are making to mobility, coupled with effective management of the parking area.The systematic review in this regard extensively discusses the application of AI and CV technologies to parking space allocation, but with specific focus on how they can be used within a seaport parking environment.

The review highlights the benefits AI and CV have in establishing effective and sustainable parking ecosystems. Lee and colleagues have assessed various studies in order to identify some of the main benefits of these technologies, which include optimized mobility, environmental impact reduction, and effective management of parking space allocation. In seaports, for example, space is limited, demands vary highly, and passenger vehicles and freight require dynamic adjustment in real-time. This flexibility is key in optimizing operational flow and minimizing delays. Lee categorizes the techniques from AI that are implemented in parking into two: there are deep learning algorithms like CNNs and YOLO which base their application on real-time detection and monitoring. From the review, it shows that models for parking allocation based on historical data combined with real-time image analyses are adaptive.

This further opens up the possibility of its use in more expansive smart city applications due to offering information feed which may be able to support the usage of traffic management and environmental monitoring systems. The results provided are a systematic review, consolidating knowledge about AI and CV application in parking but also giving calls for further research into integrating these technologies with IoT networks and automated decision-making systems to improve urban mobility. Vision-Based Deep Learning for Parking Slot Detection.

Each of the categories applies specific techniques from deep learning toward parking slot detection, thereby bringing forth both the feasibility and limitations of each technique. Object detection methods, like YOLO and Faster R-CNN, focus on identifying vehicles within predefined regions and can offer fast processing times suitable for real-time applications. However, these methods may struggle with partial occlusions, where only part of a vehicle is visible.

Datasets provided a benchmark to test and compare how certain models would work. Researchers could test their systems under different conditions, whether weather conditions or illuminations. The study therefore recommends future studies focus on the development of hybrid models that would allow the strengths of several deep learning techniques for specific challenges in parking slot detection, such as occlusions and variable lighting conditions.

### Computer Vision in Automated Parking Systems

Markus Heimberger and his co-authors discuss the use of computer vision algorithms in the design and implementation of automated parking systems. The paper goes into detail about camera system application for 3D reconstruction, parking slot marking recognition, freespace detection, and vehicle/pedestrian detection. Stereo cameras and depth sensors generate three-dimensional data that map the surrounding area, allowing for the identification of obstacles, parking slots, and their dimensions. The ability to create such a detailed model is essential for guiding vehicles into designated parking spaces while avoiding collisions. Recognizing parking slot markings is another critical functionality discussed in the paper. Using edge detection and segmentation techniques, computer vision algorithms identify lines and boundaries that denote parking spaces.

Free space detection refers to identifying unoccupied areas suitable for parking. This involves analyzing the visual field to distinguish between open spaces and obstacles, such as parked cars, pedestrians, or curbs. Vision-based algorithms, often augmented by machine learning models, classify regions as either “free” or “occupied.”

Challenging and solving the problems of developing robust automotive parking systems These systems enable vehicles to perceive autonomously detect, interpret, and respond to their environment, which is vital for achieving efficient and safe automated parking.

### Smart Parking Solutions Using Deep Learning

Below are the smart parking solutions using computer vision and deep learning algorithms for real-time monitoring of parking spaces by CVEDIA. They use pre-installed cameras to identify vehicle class within the predefined slots even during challenging weather conditions. That has eradicated the otherwise costly sensor-based method.

These examples represent tremendous advancements in AI-augmented parking systems, as well as the myriad ways applied to the core challenges faced by effective parking management. Over their components-including computer vision, machine learning, IoT devices, and user-centric design-these systems are opening up the doors to future innovation and should further enhance efficiency, accuracy, and quality user experience at urban parking.

### Deep Reinforcement Learning for Smart Parking

It is documented that a research paper has been published by Springer that states a DRL-based framework for an Industrial Internet of Things enabled intelligent parking system.

This framework involves smart cameras, fog nodes, and a cloud server using DRL to classify vehicles and then allocate vacant parking slots reasonably. This research paper presents a study where the proposed IIoT components integrate with a hierarchical computing structure extending from the edge to the cloud. This system captures real-time video feeds from near parking lots or garages and sends them to be processed at fog nodes with close to zero latency. Indeed, those intermediary devices between the edge and the cloud, fog nodes manage immediate processing and decision-making locally to avoid overloading the cloud server. Data that cannot immediately be processed at the fog nodes is forwarded to the cloud server where large volumes are analyzed and are consistently used for training the DRL model. This multi-layered architecture further boosts both speed and scalability in the parking system. AI Enabled Car Parking Using OpenCV.

Kathija962 demonstrates an AI-enabled car parking system working on GitHub, showing how the system can use OpenCV for real-time video capture taken by the parking lot. Such a system can determine the occupancy status of each parking space.This is an actual case of practical application of AI-enabled car parking using OpenCV to help in image processing.The system captures real-time video footages of a parking area and analyzes them for monitoring the occupancy status in each parking spot. OpenCV, an open-source computer vision library, is basically the basic function on which this project relied to permit functions in image processing, feature detection, and motion tracking. The system will identify vehicles with the help of a set of algorithms, picking out characteristic features and car-like shapes, and then decide to classify each parking lot as occupied or vacant. Functions in contour detection in OpenCV are particularly useful here, so the system might easily define the limits within video frames and distinguish the vehicle from the background environment. This project is a good example of how simple yet powerful computer vision tools can be leveraged effectively for creating an effective management system for parking.

It is very valuable for real time monitoring of parking occupancy in urban centres as it helps to reduce congestion and proper usage of space. The specific use of OpenCV for this particular project also highlights the importance of being modular and flexible frameworks which are adaptable to different kinds of lot layout and states. This flexibility afforded by OpenCV allows developers to tweak the system to account for factors like lighting variations or occlusions, which otherwise would interfere with the precision of the system. This project further points out how cost-effective OpenCV is for AI-powered parking solutions, as it is open-source and available to developers, thus making it more acceptable for either small-scale implementations or larger systems with scalability requirements.

## 2.3 PROPOSED SYSTEM

The proposed SmartPark AI-Enhanced Parking System involves a self-governing intelligent approach to the parking management system. The system integrates computer vision in real time with the help of predictive analytics and natural language processing for user interaction. It addresses the inefficiencies in most traditional methods of parking: it accurately detects available spaces and predicts parking demand ahead of time. Users are offered an intuitive interface that lets them search and reserve parking spots. Objectives: Smartpark is said to reduce search times, optimize parking resource allocation, and alleviate traffic congestion. It aspires to give people a seamless parking experience and cater to the needs of urban drivers and facility operators alike.

The three major modules that the core architecture of SmartPark is divided into include the Computer Vision Module, Predictive Analytics Module, and User Interaction Module. The Computer Vision Module is based on the use of ultralytics's YOLO v5 model combined with PIL, a deep learning framework optimized for real-time object detection, thereby it is fast and accurate. This module performs real-time video analysis to identify available parking spots.

It would therefore be able to identify which of the spots are open or occupied by processing live feeds coming from cameras. YOLO can detect objects and classify areas as either "vacant" or "occupied" with a high accuracy level. The module reports this information. Simultaneously, it can send updates to other modules, such as a central dashboard or user interface.

Using the Python Imaging Library for image preprocessing, the module's analysis is very fast, taking minimal latency while still allowing for accuracy in detection. With such real-time capabilities, SmartPark can provide live updates on parking space availability. This can help drivers make quick and well-informed decisions.

The Predictive Analytics Module shall look into historical occupancy and identify trends over time to predict parking demand. Supervised learning models are used to inform estimation of the future availability in terms of peak hours, seasonality of demand, and specific location usage. Through data normalization techniques, machine learning models are ensured to work well on different datasets, hence providing reliable insights that can support users to plan their parking well ahead. These features improve the user experience but also benefit facility managers in terms of better utilization of parking resources, perhaps by adjusting space allocations or promoting demand-based pricing related to estimated levels of usage.

The User Interaction Module primarily focuses on making the system accessible and user-friendly. It integrates with NLP libraries, which include spaCy and NLTK. These tools grant the module a chatbot interface from which users can merely send voice or text inputs to SmartPark.

This chatbot shares real-time information about parking, gives directions to available open spots, and reservation options-all of which is done to ensure the delivery of a seamless, hands-free experience for users on-the-go. It is available through vehicle infotainment systems or mobile devices, thereby making sure that responses are received immediately and appropriately to navigate parking facilities and reserve spots when needed. The module enables it to understand and provide a lot of different queries made by the users, which enhances engagement and convenience for drivers.

Overall, the proposed SmartPark is the scalable, adaptable parking solution for deployment in urban environments. The modular architecture thus allows integration with other smart city systems as well as traffic management and IoT networks in support of a more connected and efficient urban infrastructure. Reducing the times drivers spend hunting for parking through SmartPark improves the satisfaction of users, reduces congestion and emissions, and goes on to promote a more sustainable urban ecosystem. It is indeed because of this combination of real-time detection, predictive analytics, and guidance that SmartPark stands as the ultimate comprehensive and future-proof solution in modern parking management.

## 2.6 PRACTICAL STUDY

### Introduction

The ever-increasing challenges of parking with urbanization have led to congestion, wastage of time as well as fuel. SmartPark AI-Enhanced Digital Parking System takes its roots from modules of computer vision, machine learning, and user-interaction. This technical, economic, operation, and scalability-based study evaluates the practical use of SmartPark.

### Technical Feasibility

SmartPark consists of three essential modules: Computer Vision, Predictive Analytics, and User Interaction.

**Computer Vision Module:** This module utilizes YOLO v5, an advanced object detection model, to identify parking spots in real time by performing video analysis. The efficiency of the YOLO v5 model allows it to process camera feeds quickly, identifying spots as vacant or occupied with a high degree of accuracy. PIL supports image preprocessing to help enhance the accuracy of the detection. Given the maturity of YOLO v5 in real-time application use, this module is technically feasible, and with reliable performance in urban settings, especially well-lit parking areas.

**Predictive Analytics Module:** Using python-based supervised ML algorithms, this module predicts parking demand based on historical data. Data normalization is important in the absence of which occupancy variations can skew the predictions of models. Forecasting of trends, it reduces resource wastage at peak hours. The technical requirements are reasonably achievable, given existing cloud solutions dealing with storage of historical data and computational power to work out real-time predictions.

**User Interaction Module:** The module can be suggested as NLP-based chatting, where it can share availability and directions with the infotainment system of the vehicle with users. Libraries like spaCy or NLTK are enabled to create such smooth interactions, which enhance user experience. The feasibility of the module is quite high since technologies for such a chatbot type are already used in most applications.

**Operational Feasibility**

Implementing SmartPark requires coordination with local governments, parking facility owners, and technology providers. The system's operations involve continuous data collection, processing, and real-time updates, necessitating stable network connectivity and reliable camera infrastructure. User adoption will depend on ease of use and accessibility, which the chatbot module aims to address by integrating seamlessly with existing vehicle infotainment systems.

The potential challenge to operational feasibility is data privacy, as the system has to include sensitive user data: vehicle location and identities of users. Strict privacy policies and data encryption can mitigate these concerns, ensuring compliance with data protection regulations.

**Scalability**

SmartPark design is scalable and modular to suit urban landscapes. It could thus be first deployed in highly trafficked spots like city centers or shopping malls, then expanded where needed. Involving other smart city programs, such as traffic management or public transportation, would add to its long-term viability. Long-term plans may include predictive analytics to track surge demand due to events or dynamic pricing aligned with real-time demand patterns.

# CHAPTER 3

# SYSTEM DESIGN & ANALYSIS

## 3.1 PROJECT PERSPECTIVE

**Introduction**

The SmartPark project aims to solve the critically important problem of city parking management using a holistic AI-enabled system that integrates real-time detection, predictive analytics, and user interactivity. SmartPark makes use of sophisticated technologies transformed to mean efficient, user-friendly, and scalable for the needs of future urbanization.

**Objectives and Purpose**

The main objective of SmartPark is to reduce congestion, lessen the time spent by drivers looking for parking spaces, and optimize the utilization of parking space available. These objectives are addressed through three major components: Computer Vision Module, which identifies open parking spots from live video feeds; Predictive Analytics Module, which uses past trends to predict availability in advance; and User Interaction Module, an intuitive chatty interface to inquire about parking status or seek directions to a parking location. This architecture should all ease the process of parking, save time, and reduce emissions from vehicles for cleaner, more efficient cities.

**Target Users**

The system is targeted for urban drivers, city municipalities, and private parking operators. For the driver, SmartPark simplifies parking and even provides space prebooking through an infotainment-compatible interface. City planners and parking authorities enjoy better traffic flow and insights into parking patterns, with potential resource reallocation by using real-time data on occupancy levels.

**Technological Framework**

Architecture in SmartPark is developed based on tested AI tools and frameworks. YOLO v5 is fast for object detection and real-time analysis, whereas machine learning algorithms process historical data to accurately forecast parking demand. The chatbot empowered through Python NLP libraries like spaCy and NLTK interacts with the users. The combined effect of these technologies gives a reliable, scalable, and compatible solution that can work within all types of different and complex urban spaces, from private parking lots.

### Project Scope and Impact

The scope encompasses fast dividends that include saving time for parking, more satisfaction on behalf of the users, and lower traffic congestion. In the long run, the initiative could be integrated with higher smart city actions, connectedness, public transport system, traffic management, and green urban planning. SmartPark could also contribute toward sustainability objectives by optimizing its parking lots and lessening the idling time, which would ultimately have positive impacts in reducing emissions and have a lower carbon footprint.

### Support for Dynamic Pricing and Resource Allocation

By leveraging real-time and predictive data, SmartPark can assist parking operators in implementing dynamic pricing strategies. This encourages optimal usage of underutilized spaces during off-peak hours and ensures fair resource allocation during peak demand. City planners can also use these insights to identify high-demand zones and plan additional infrastructure or alternative transport solutions.

### Enhanced Urban Mobility

SmartPark contributes to smoother urban mobility by reducing bottlenecks caused by vehicles searching for parking. This, in turn, improves traffic flow and decreases the likelihood of accidents caused by congestion or sudden stops. The system's predictive analytics also help reroute drivers to less congested areas, ensuring balanced usage of available parking resources.

### Sustainability and Environmental Benefits

The reduction in vehicle idling time directly translates to lower greenhouse gas emissions, contributing to cleaner urban environments. By optimizing parking management, SmartPark aids in reducing the urban heat island effect often exacerbated by overcrowded parking lots. Furthermore, integrating the system with electric vehicle (EV) charging stations can promote the adoption of EVs by ensuring drivers have access to both parking and charging facilities.

## 3.2 SYSTEM ARCHITECTURE

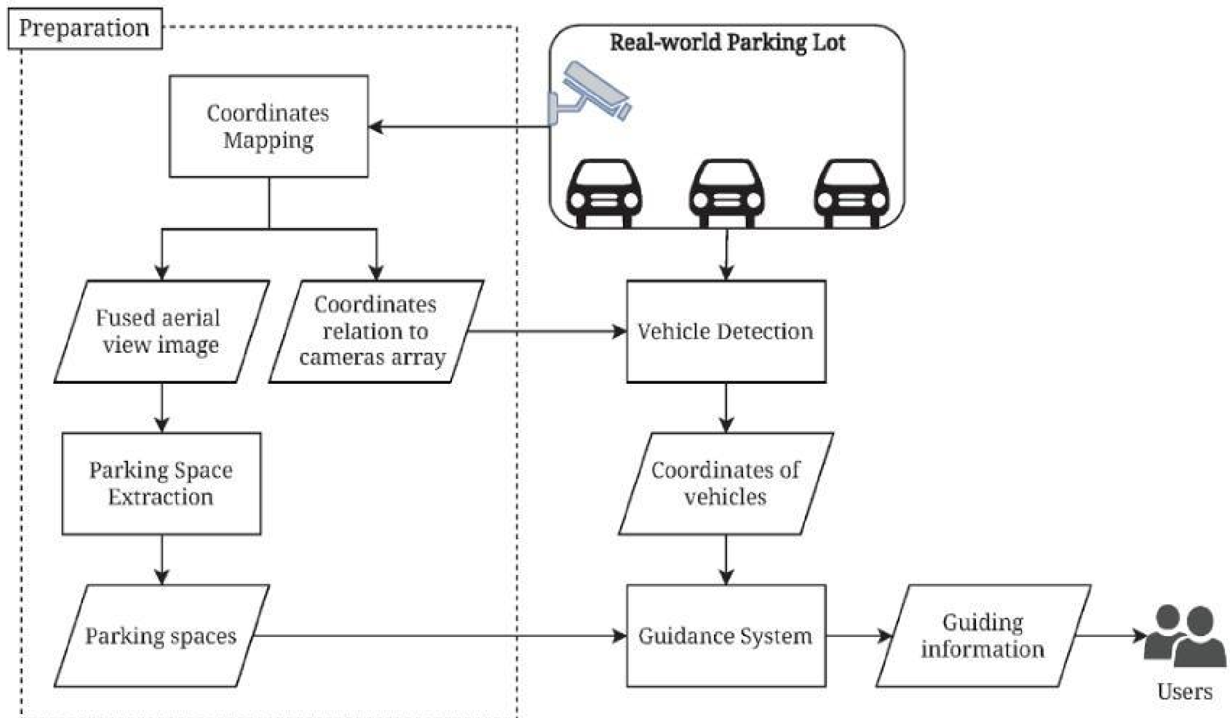



**Figure - 1**

The Smart Parking System architecture consists of the following key stages:

**1) Preparation Phase:**

i. **Coordinates Mapping:** Maps parking lot and camera positions.
ii. **Parking Space Extraction:** Identifies and labels each parking space on the aerial image.

**2) Real-Time Monitoring:**

i. **Vehicle Detection**: Use Computer Vision techniques to detect vehicles in the parking lot.
ii. **Coordinate Mapping**: Maps vehicle positions to identify occupied spaces.

**3) Detection Model:**

i. Uses YOLO (You Only Look Once) and Python for object detection.
ii. Integrates with CCTV cameras to monitor parking lot areas.

**4) Object Detection:**

i. Identifies vehicles and vacant parking spaces in real-time.
ii. Detects vehicles entering and exiting the parking space.

**5) User Interaction**:

i. Delivers real-time parking info and directions to users through an interface.
ii. This system provides automated, real-time parking guidance and availability updates to users.

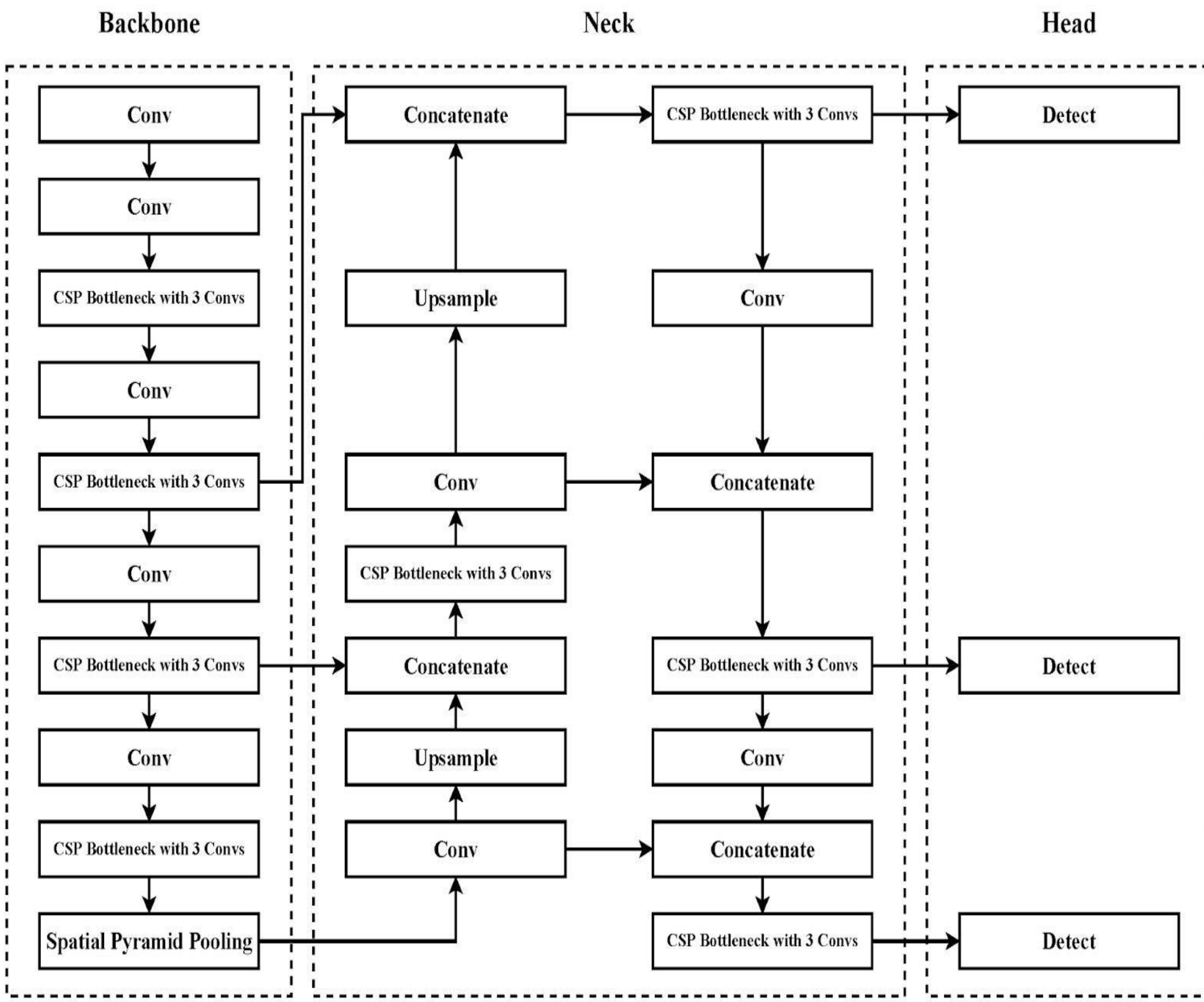



**Figure - 2**

**Overview of Structure of YOLO v5**

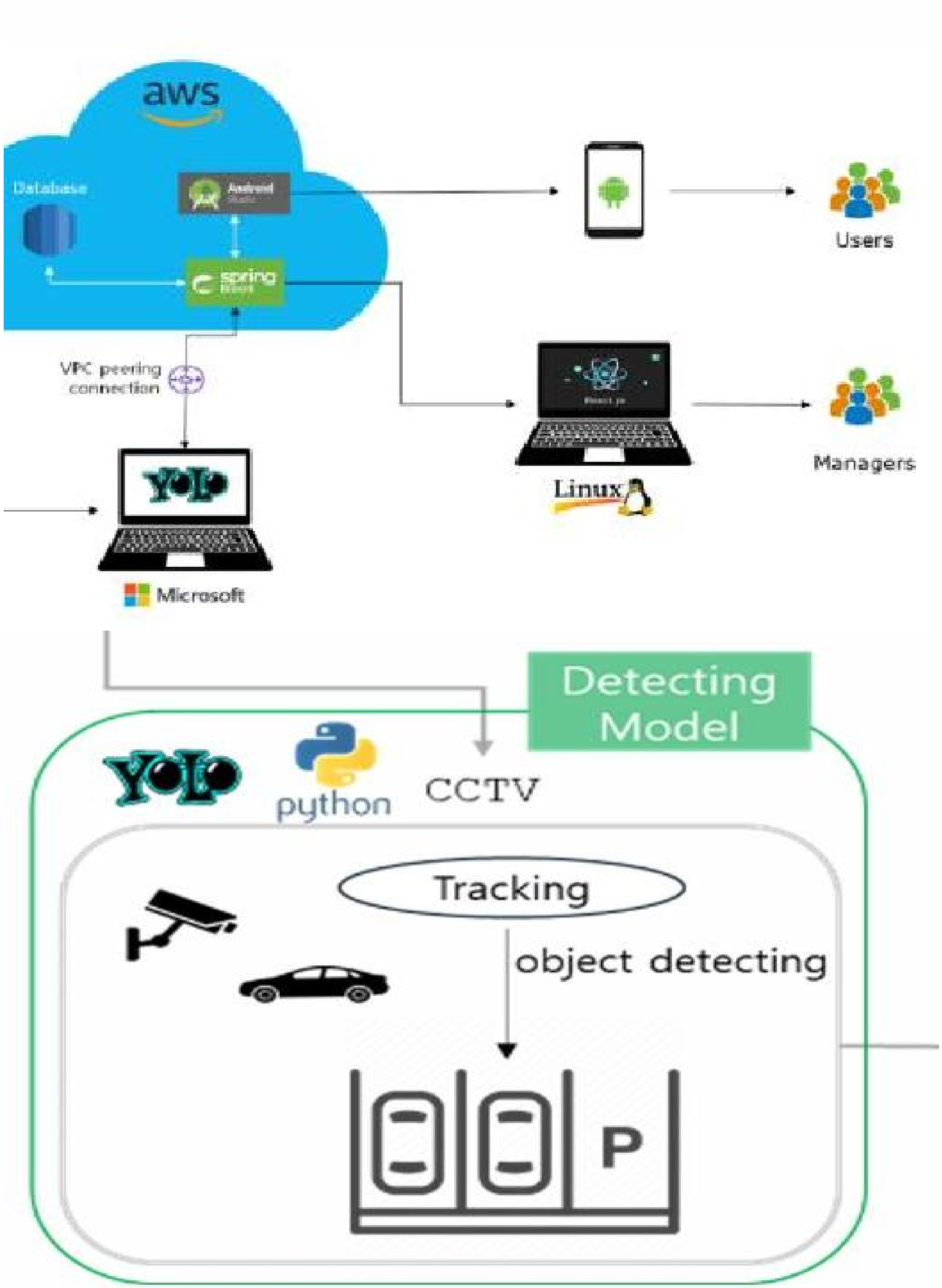



**Figure - 3 - YOLO Detection**

## 3.3 MODELS USED

### 1. Computer Vision Module

The Computer Vision module is in charge of the real-time detection of available parking spots. This module primarily depends on YOLO v5 for object detection and several other support libraries for image processing and handling.

### YOLO v5 (You Only Look Once version 5)

i. **Description:** YOLO is a family of deep learning models optimized for real-time object detection, which uses version 5 for speed and accuracy.
ii. **Developed By:** Ultralytics - an organization dedicated to delivering solutions in AI technology, with special proficiency in object detection, and computer vision.
iii. **Functionality:** YOLO v5 can run its entire computation in one pass, which allows it, in turn, to spot several objects in one go, making the application very efficient in real time. For instance, for the SmartPark application, it is used in checking video feeds taken from parking lots by determining open and occupied parking spots.

**Frameworks and Tools:** The model leverages PyTorch, a popular deep learning framework, for implementation, training, and inference. PyTorch allows for efficient computation on both CPUs and GPUs, making YOLO v5 suitable for deployment in environments with limited resources.

The main equations focus on bounding box predictions, confidence scores, and loss functions used in training.

### Bounding Box Prediction

YOLO predicts bounding boxes around detected objects. Each box is represented by:

(x,y)(x, y)(x,y): center of bounding box

w: width of bounding box

h: height of bounding box

Intersection over Union (IoU)

YOLO calculates the IoU between the predicted and ground truth boxes to score the accuracy of detection:

$$IoU = \text{Area of Overlap} / \text{Area of Union}$$

where:

Area of Overlap is the area of intersection between the predicted box and the ground truth box.

Area of Union is the total area covered by both boxes.

**Supporting Libraries**

PIL (Python Imaging Library): Such a preprocessing task as resizing, cropping, format conversion, etc., is provided by the PIL library, available under the Pillow package. All these preprocessing steps are necessary to ensure that images fed into YOLO are optimized for accurate detection.

**Image Resizing**

Image Resizing stretches/shrinks the image size in terms of a scaling factor. Given a scaling factor sss, the new width W′ and height H′ of an image are:

$W' = s \cdot W, H' = s \cdot H$ ---- eq 1

where WWW and HHH represent the original width and height.

**Convolution for Image Filtering**

For effects like blurring or sharpening, PIL uses convolution. An image I is convolved with a filter K to produce an output O:

$O(x,y) = i=-k\sum k \quad j=-k\sum k \quad K(i,j) \cdot I(x-i,y-j)$ ----- eq 2

where :

K(i,j) represents the filter kernel.

I(x, y) represents the value of a pixel at pixel location (x, y)

k is the kernel size.

**OpenCV:** Often utilized with PIL, OpenCV is computer vision library that provides varieties of tools for processing images and video. In SmartPark, OpenCV is used to capture and manipulate live video feeds which allows YOLO to work better for detecting parking spots. Also useful in the transformations such as rotation and scaling on images which also enhance the robustness of the model.

**Ultralytics Hub:** It is the model deployment and performance monitoring platform provided by Ultralytics. Such integration with Ultralytics Hub will ensure SmartPark receives real-time model updates, monitoring of performance, and even management of the model.

**2. Predictive Analytics Module**

Predictive analytics module uses past data to predict available parking space. This capability is built using supervised learning models and a range of data-handling libraries. Supervised Learning Models

**Linear Regression:** It is a simple supervised learning model that creates a linear relationship between past occupancy rates and predicted future availabilities. Easy to interpret, it forms a basic approach to any predictive analytics.

**Data Handling and Processing Libraries**

i. **NumPy:** It's a library used for numerical operations, normalization of the data, array manipulation, etc. - ensures efficient handling of large data sets, especially when preparing historical data for predicting models.
ii. **Pandas:** it is the core library for data manipulation and storage at Python level. It becomes handy while dealing with the history data being fed to train and validate the model.
iii. **Scikit-Learn:** Amongst libraries for machine learning, this is used in applying supervised learning models such as Linear Regression, Random Forests, Gradient Boosting, etc. It has extra methods to preprocess data through normalization and scaling, which increase performance of the model.
iv. **Statsmodels:** The package is used to undertake statistical models, like ARIMA models, in time series forecasting. It has tools for both the fitting and evaluation of predictive models to enhance the accuracy of availability predictions.

**Prediction and Evaluation Techniques**

Cross-Validation: Cross-validation works by using historical data to partition the training and testing sets while evaluating the predictive models' accuracy. In this case, the technique ensures the model generalizes well to new data.

### 3. User Interaction Module

The User Interface module would enable users to interact with the system through a chatbot interface. This module would deploy NLP tools reading and responding to queries by users about parking availability, directions, and potential reservations.

#### Natural Language Processing Libraries

i. **spaCy:** This application uses spaCy as an efficient Python library supported by multiple high-performance applications. It is implemented to identify intent from the user query and the appropriate response to the question that may pertain to parking availability or the directions. Besides this, the features of named entity recognition and POS tagging elevate the power of a chatbot in terms of contextual understanding.
ii. **NLTK Natural Language Toolkit):** NLTK is yet another popular NLP library that provides a suite of text processing tools, for tokenization, stemming, and text classification. NLTK is combined with spaCy to enrich the functionalities of a chatbot to be able to understand more extensive user inputs.

#### Tokenization

Tokenization refers to splitting the text into individual units called tokens. Let SSS be a sentence formed by words (w1,w2,…,wn). Tokenization can be defined as:

$S \rightarrow \{w1,w2,\ldots,wn\}$ ---- eq 3

Every word wiw_iwi in the sentence SSS is considered as individual that can be further processed by following NLP processing.

### 4. File Handling and System Libraries

There are various system libraries utilized for file handling, data management, and resource management in SmartPark. They are-

i. **shutil:** The shutil module in Python provides a package of functions that involve high-level file operations. These are mainly used for moving, copying, and deleting files in the Smart-Park system to handle data as seamlessly as possible.
ii. **OS:** OS library must to communicate with os, when dealing with paths for files and directory structures to access system-level data and control over file

organization, especially when working with storage for models, log files, and configuration management.

iii. **glob:** Using the glob library easily matches file pattern. Great when dealing with many video files or pictures inside Computer Vision Module.

iv. **Pathlib:** Makes easy usage of file paths. This package includes easier file handling, especially in Windows, Unix-like systems, etc. Pathlib also uses object-oriented methods to deal with paths to files, so more flexible operations on files are achievable.

**5. Additional Libraries and Tools Optimization**

Logging: Python's native logging tool, which is used to trace events, errors or operational status, in the Smart-Park system.

## 3.4 METHODOLOGY

The methodology of the Smart-Park AI-Enhanced Parking System procedure follows a series of steps: data pre-processing, model training, and splitting the data. It employs computer vision for live predictions.

**1. Data Pre-processing and Cleaning:**

Image File Validation and Deletion: In the first step before processing, the system verifies whether a corresponding .txt file with label data exists for every image. If there isn't a corresponding .txt file for an image, the image is deleted to ensure uniformity in the training and validation data.

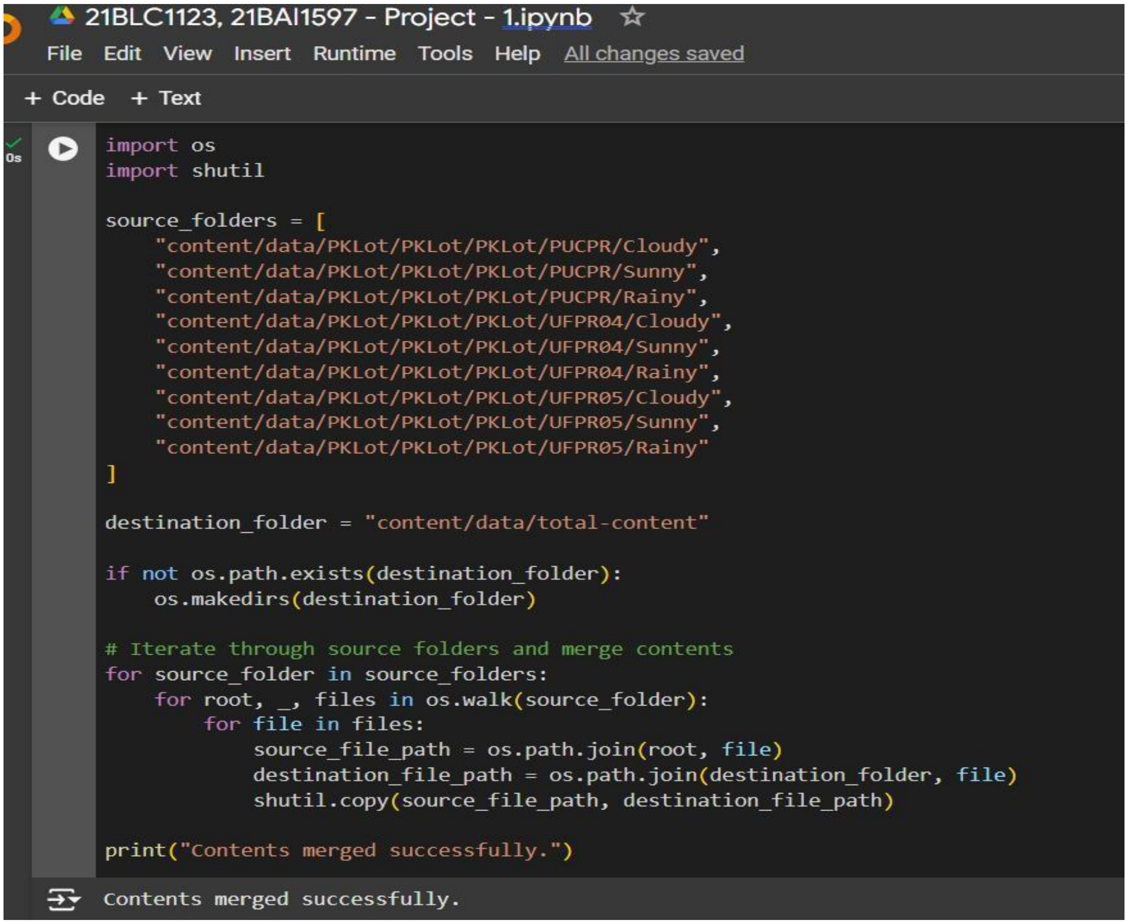
```
import os

# remove the image that has no corresponding txt file

file_path = "content/data/total-content/2012-11-06_18_48_46.jpg"
try:
    os.remove(file_path)
    print(f"File '{file_path}' has been removed successfully.")
except OSError as e:
    print(f"Error removing the file '{file_path}': {e}")
```

**Figure - 4 -** Input Code

**Merge all folders into one.**




```python
import os
import shutil

source_folders = [
    "content/data/PKLot/PKLot/PKLot/PUCPR/Cloudy",
    "content/data/PKLot/PKLot/PKLot/PUCPR/Sunny",
    "content/data/PKLot/PKLot/PKLot/PUCPR/Rainy",
    "content/data/PKLot/PKLot/PKLot/UFPR04/Cloudy",
    "content/data/PKLot/PKLot/PKLot/UFPR04/Sunny",
    "content/data/PKLot/PKLot/PKLot/UFPR04/Rainy",
    "content/data/PKLot/PKLot/PKLot/UFPR05/Cloudy",
    "content/data/PKLot/PKLot/PKLot/UFPR05/Sunny",
    "content/data/PKLot/PKLot/PKLot/UFPR05/Rainy"
]

destination_folder = "content/data/total-content"

if not os.path.exists(destination_folder):
    os.makedirs(destination_folder)

# Iterate through source folders and merge contents
for source_folder in source_folders:
    for root, _, files in os.walk(source_folder):
        for file in files:
            source_file_path = os.path.join(root, file)
            destination_file_path = os.path.join(destination_folder, file)
            shutil.copy(source_file_path, destination_file_path)

print("Contents merged successfully.")
```

```
Contents merged successfully.
```

**Figure - 5 -** Input Code

- **Modify labels:** We also need to modify the labels to be in the same format as YOLOv8 requires. Our labels right now are in xml format, YOLOv8 need it in a txt file with the following format:

  class x_center y_center width height

- The next cell will convert the xml labels to txt labels and move the xml files since we don't need them.The image_height and image_width need to be the height and the width of the image (as obvious as that sounds). This is because when converting from xml, we want to convert the bounding box coordinates to be relative to the image size. This is done by dividing the bounding box coordinates by the image size. This is also why we need to know the image size. (1280x720 in this case).

```python
import xml.etree.ElementTree as ET

input_folder = "content/data/total-content"
output_folder = "content/data/labels-xml"

image_width = 1280
image_height = 720

class_mapping = {"1": 1, "0": 0}

if not os.path.exists(output_folder):
    os.makedirs(output_folder)

xml_files = [f for f in os.listdir(input_folder) if f.endswith(".xml")]
for xml_file in xml_files:
    xml_path = os.path.join(input_folder, xml_file)

    tree = ET.parse(xml_path)
    root = tree.getroot()

    txt_filename = os.path.splitext(xml_file)[0] + ".txt"
    txt_path = os.path.join(output_folder, txt_filename)

    with open(txt_path, "w") as txt_file:
        for space in root.findall("space"):
            occupied = space.get("occupied")
            class_index = class_mapping.get(occupied, -1)

            if class_index == -1:
                continue

            rotated_rect = space.find("rotatedRect")
            center = rotated_rect.find("center")
            size = rotated_rect.find("size")

            center_x = float(center.get("x"))
            center_y = float(center.get("y"))
            width = float(size.get("w"))
            height = float(size.get("h"))

            x_center = center_x / image_width
            y_center = center_y / image_height
            w = width / image_width
            h = height / image_height

            txt_file.write(f"{class_index} {x_center:.6f} {y_center:.6f} {w:.6f} {h:.6f}\n")

    new_xml_path = os.path.join(output_folder, xml_file)
    os.rename(xml_path, new_xml_path)

print("Annotations generated and XML files moved.")
```

```
Annotations generated and XML files moved.
```

**Figure - 6 -** Input Code

**2) Data Splitting for Training, Testing, and Validation**

**Setup and Folder Creation**: To facilitate organized data handling, separate folders are created for training, testing, and validation datasets (`train`, `test`, `val`). If these folders don't already exist, they are created to avoid file organization errors.

```
import random
import os

# remove the image that has no corresponding txt file

file_path = "content/data/total-content/2012-11-06_18_48_46.jpg"
try:
    os.remove(file_path)
    print(f"File '{file_path}' has been removed successfully.")
except OSError as e:
    print(f"Error removing the file '{file_path}': {e}")


source_folder = "content/data/total-content"
train_folder = "content/data/train"
test_folder = "content/data/test"
val_folder = "content/data/val"

train_ratio = 0.7
test_ratio = 0.15
val_ratio = 0.15

for folder in [train_folder, test_folder, val_folder]:
    if not os.path.exists(folder):
        os.makedirs(folder)
```

**Figure - 7 -** Input Code

```
all_files = os.listdir(source_folder)
image_files = [f for f in all_files if f.endswith(".jpg")]

# Calculate the number of samples for each split
num_samples = len(image_files)
num_train = int(train_ratio * num_samples)
num_test = int(test_ratio * num_samples)
num_val = num_samples - num_train - num_test

random.shuffle(image_files)

train_files = image_files[:num_train]
test_files = image_files[num_train:num_train + num_test]
val_files = image_files[num_train + num_test:]
```

**Figure - 8 -** Input Code

**File Selection and Shuffling**: All .jpg files are selected and shuffled to ensure random sampling for training, testing, and validation. The data is split into proportions based on the specified ratios.

### 3) Image and Label Organization

Folder Structure for Model Training: Within each dataset folder (train, test, val), subfolders images and labels are created to separate image files and label files, aligning with standard practices for YOLO model training.

```
datasets = ['train', 'val', 'test']
source_folder = 'data'

for dataset in datasets:
  dataset_folder = os.path.join(source_folder, dataset)

  # Create "images" and "labels" folders
  images_folder = os.path.join(dataset_folder, "images")
  labels_folder = os.path.join(dataset_folder, "labels")
  os.makedirs(images_folder, exist_ok=True)
  os.makedirs(labels_folder, exist_ok=True)

  # Organize image and label files
  for file in os.listdir(dataset_folder):
    if file.endswith(".jpg"):
      image_path = os.path.join(dataset_folder, file)
      image_destination = os.path.join(images_folder, file)
      shutil.move(image_path, image_destination)
    elif file.endswith(".txt"):
      label_path = os.path.join(dataset_folder, file)
      label_destination = os.path.join(labels_folder, file)
      shutil.move(label_path, label_destination)

print("Datasets organized successfully.")
```

```
Datasets organized successfully.
```

**Figure - 9 -** Input Code

### 4) Model Prediction and Object Detection

Random Image Selection and Model Prediction: A random image from the test dataset is selected and passed to the trained model for prediction. This step is critical to validate the model's detection accuracy in identifying available parking spots.

```
from PIL import Image

test_images_directory = "./data/test/images"

image_files = [f for f in os.listdir(test_images_directory) if f.lower().endswith(".jpg")]

random_image_filename = random.choice(image_files)
random_image_path = os.path.join(test_images_directory, random_image_filename)

results = model.predict(random_image_path)
```

**Figure - 10 -** Input Code

**5) Bounding Box Extraction:** After prediction, bounding boxes are extracted for each detected object. For each bounding box, object class, coordinates, and confidence score are recorded.

```
[9] for box in result.boxes:
      label = result.names[box.cls[0].item()]
      coords = [round(x) for x in box.xyxy[0].tolist()]
      prob = round(box.conf[0].item(), 4)
      print("Object: {}\nCoordinates: {}\nProbability: {}".format(label, coords, prob))
```

**Figure - 11 -** Input Code

### 6) Post-Prediction Folder Organization

- Organizing Image and Label Files: The `images` and `labels` folders in each dataset are further organized for efficient model training, separating image files from their label files.

```
from PIL import Image

random_image_path = "/content/img 5.png"
results = model.predict(random_image_path)

result = results[0]
for box in result.boxes:
  label = result.names[box.cls[0].item()]
  coords = [round(x) for x in box.xyxy[0].tolist()]
  prob = round(box.conf[0].item(), 4)
  print("Object: {}\nCoordinates: {}\nProbability: {}".format(label, coords, prob))
```

**Figure - 12 -** Input Code

```
image 1/1 /content/img 5.png: 480x640 20 emptys, 8 occupieds, 1036.3ms
Speed: 5.1ms preprocess, 1036.3ms inference, 1.6ms postprocess per image at shape (1, 3, 480, 640)
Object: occupied
Coordinates: [608, 663, 710, 834]
Probability: 0.8858
Object: empty
Coordinates: [350, 715, 538, 806]
Probability: 0.8702
Object: empty
Coordinates: [284, 208, 342, 274]
Probability: 0.8633
Object: occupied
Coordinates: [82, 318, 169, 408]
Probability: 0.8557
Object: occupied
Coordinates: [678, 577, 767, 744]
Probability: 0.8554
Object: occupied
Coordinates: [715, 487, 789, 638]
Probability: 0.8456
Object: empty
Coordinates: [231, 241, 297, 320]
Probability: 0.8453
Object: empty
Coordinates: [432, 176, 486, 246]
Probability: 0.845
Object: empty
Coordinates: [338, 244, 398, 326]
Probability: 0.8431
Object: empty
Coordinates: [335, 182, 391, 243]
Probability: 0.8407
Object: empty
Coordinates: [475, 148, 519, 217]
Probability: 0.8385
Object: occupied
Coordinates: [0, 380, 90, 473]
```

**Figure - 13 -** Input Code

```
Probability: 0.835
Object: empty
Coordinates: [163, 278, 236, 361]
Probability: 0.8336
Object: empty
Coordinates: [285, 292, 349, 362]
Probability: 0.8266
Object: empty
Coordinates: [377, 154, 433, 214]
Probability: 0.8244
Object: occupied
Coordinates: [577, 388, 712, 441]
Probability: 0.8154
Object: empty
Coordinates: [214, 334, 291, 419]
Probability: 0.8147
Object: empty
Coordinates: [639, 323, 762, 375]
Probability: 0.8131
Object: occupied
Coordinates: [756, 411, 822, 547]
Probability: 0.8066
Object: empty
Coordinates: [769, 342, 827, 467]
Probability: 0.8003
Object: occupied
Coordinates: [809, 290, 865, 405]
Probability: 0.7946
Object: empty
Coordinates: [394, 211, 451, 283]
Probability: 0.7944
Object: empty
Coordinates: [699, 240, 812, 281]
Probability: 0.7731
Object: empty
Coordinates: [638, 281, 770, 332]
Probability: 0.7572
Object: empty
Coordinates: [856, 202, 898, 309]
Probability: 0.7562
Object: empty
Coordinates: [831, 237, 877, 353]
Probability: 0.7451
Object: empty
Coordinates: [878, 168, 916, 269]
Probability: 0.7406
Object: empty
Coordinates: [708, 205, 811, 243]
Probability: 0.7184
```

**Figure - 14 -** Input Code

This methodology outlines the entire process, from data cleaning to prediction, and file organization. Each step is essential to ensure the system's reliability in accurately detecting parking spots and efficiently organizing data for future training and validation.

```python
nltk.download('punkt')
nltk.download('stopwords')

# Initialize stop words
stop_words = set(stopwords.words('english'))

# Define the chatbot's intents and responses
responses = {
    "availability": [
        "There are several spots available right now.",
        "It looks like parking is almost full, but there are a few spots left.",
        "Unfortunately, parking is currently full. Try checking again later."
    ],
    "booking": [
        "To reserve a spot, please provide your details, such as name and the parking time.",
        "Booking a spot requires your contact information and estimated arrival time."
    ],
    "greeting": [
        "Hello! How can I assist you with parking today?",
        "Hi there! Need help finding a parking spot?",
        "Welcome! I'm here to help you with parking availability and booking."
    ],
    "goodbye": [
        "Thank you for using the parking assistant. Safe travels!",
        "Goodbye! Feel free to reach out if you need parking help again.",
        "Take care! Hope you find the perfect parking spot."
    ],
    "unknown": [
        "I'm sorry, I didn't quite get that. Could you rephrase?",
        "Can you clarify? I'm here to help with parking availability and bookings."
    ]
}

# Define a function to classify the user input based on keywords
def classify_intent(user_input):
    user_tokens = word_tokenize(user_input.lower())
    filtered_tokens = [word for word in user_tokens if word not in stop_words]

    # Simple keyword matching for intent classification
    if any(word in filtered_tokens for word in ["available", "availability", "spot", "space"]):
        return "availability"
    elif any(word in filtered_tokens for word in ["book", "reserve", "reservation"]):
        return "booking"
    elif any(word in filtered_tokens for word in ["hello", "hi", "hey"]):
        return "greeting"
    elif any(word in filtered_tokens for word in ["bye", "goodbye", "later"]):
        return "goodbye"
    else:
        return "unknown"
```

**Figure - 15 -** Input Code

```
# Define a function to generate responses based on classified intent
def generate_response(intent):
    if intent in responses:
        return random.choice(responses[intent])
    else:
        return random.choice(responses["unknown"])

# Simulate booking functionality
def handle_booking():
    print("Please enter your name for booking:")
    name = input("Name: ")
    print("Please enter the time you would like to reserve the parking spot for:")
    time = input("Time: ")
    return f"Booking confirmed for {name} at {time}. Your parking spot is reserved!"

# Main loop to interact with the chatbot
def parking_chatbot():
    print("Parking Assistant: Hi! I'm here to help you with parking availability and booking.")

    while True:
        user_input = input("You: ")
        if user_input.lower() in ["exit", "quit", "goodbye", "bye"]:
            print("Parking Assistant:", generate_response("goodbye"))
            break

        # Classify the intent of the user's input
        intent = classify_intent(user_input)

        # Generate and print the response
        response = generate_response(intent)
        print("Parking Assistant:", response)

        # Additional handling if the intent is 'booking'
        if intent == "booking":
            booking_confirmation = handle_booking()
            print("Parking Assistant:", booking_confirmation)

# Run the chatbot
parking_chatbot()
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

**Figure - 16 -** Input Code

## 3.4 PERFORMANCE REQUIREMENTS

### 1. Detection and Accuracy Requirements

The computer vision module should certainly be able to detect the presence of any parking spots reliably all the time. The accuracy of video analysis with YOLO v5 and the Python Imaging Library must be optimized to tell apart an occupied from a vacant spot under all environmental conditions. The performance requirements should ensure that real-time video analysis correctly identifies at least 95 percent of the spots under different lighting conditions, viewpoints, and distances. The number of false positives, or incorrectly ascribing an occupied space as vacant, and false negatives in this case, missing an empty spot should be minimized with acceptable error rates under 5% to ensure high dependability for urban applications.

**2 Predictive Analytics Performance**

Predictive analytics, supported by Python, and supervised learning models, must predict the parking demand based on historical data. The system has to achieve at least 85% accuracy in predicting parking space availability for successive time periods so that users can make the right decision on parking availability before landing at their destination. The system should keep improving as it adapts to new patterns of changing usage in parking. This implies retraining the model at regular intervals, say every month by using the latest parking occupancy data to improve the prediction accuracy and adapt to seasonal changes, traffic movements, or local activities.

**3 Latency of the System and Real-Time Responsiveness**

Since the system will be an application in real-time, minimal latency is quite crucial. The computer vision module should process each video frame in less than 50 milliseconds, thereby providing near-instantaneous feedback on parking availability. To be fluid, especially in urban centers, which are typically congested or highly used centers, the system must allow end-to-end latency below 100 milliseconds from image capture to display on the user interface. The predictive analytics module should output its predictions within 200 milliseconds of receiving a query to give good response times about users' inquiries

**4 Interaction and Experience Requirements for Users**

The user interaction module, using natural language processing through libraries such as spa-Cy or NLTK, should enable smooth communication between the users and the system. The chatbot should have at least a 95% success rate in interpreting user queries regarding the availability of parking spaces, directions, and bookings, and accordingly give pertinent answers in real time. Putting in views about parking information and predictive analytics, it should intuitively be part of the infotainment system of the vehicle to avoid overloading the driver with accurate real time update. It should be tested for friendliness and accessibility with the various devices but possibly on screens of different sizes. Examples of visual elements include occupancy indicators, navigation prompts and should clearly be interpreted on a glance to enhance driver safety and convenience

# CHAPTER 4

# RESULT & OUTPUT

## INPUT IMAGE - (1)

**Figure - 17 -** Output Image

**Figure - 18 -** Output Image

# OUTPUT IMAGE - (1)

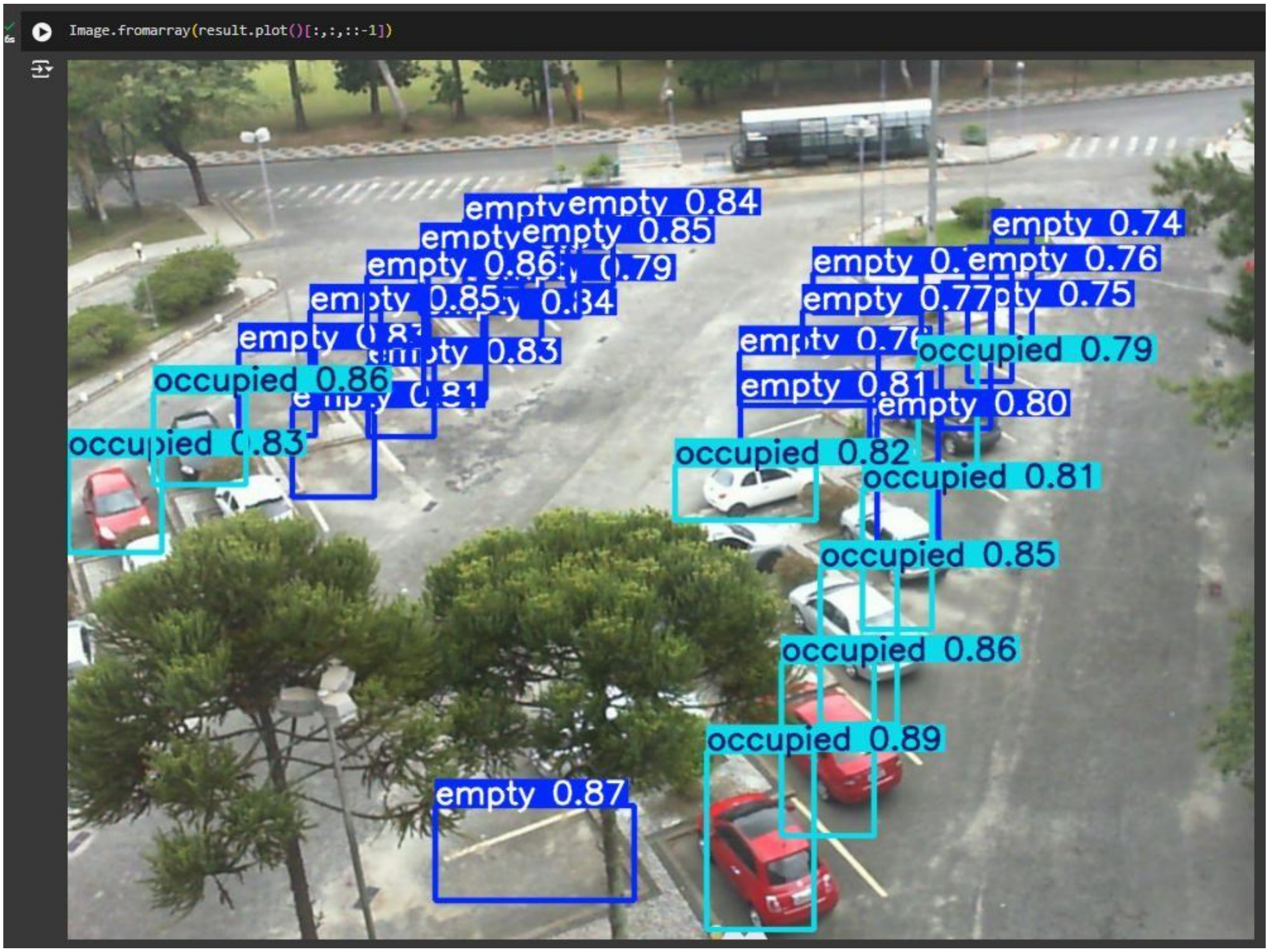



**Figure - 19 -** Output Image

```
image 1/1 /content/img 3.jpg: 384x640 20 emptys, 8 occupieds, 612.5ms
Speed: 22.8ms preprocess, 612.5ms inference, 38.7ms postprocess per image at shape (1, 3, 384, 640)
```

**Figure - 20 -** Output Image

**2)**


**Figure - 21 -** Output Image

## RESULT TABLE -

| | A | B | C |
|---|---|---|---|
| 1 | **Metric** | **Description** | **Result/Value** |
| 2 | Object Detection Accuracy | Percentage accuracy of detecting vehicles in parking spots | 95% |
| 3 | Tracking Accuracy | Accuracy in tracking vehicle movements in/out of spaces | 93% |
| 4 | False Positive Rate | Incorrect detections (e.g., detecting an empty spot as occupied) | 3% |
| 5 | False Negative Rate | Missed detections (e.g., vehicle not detected in occupied spot) | 2% |
| 6 | Occupancy Detection Accuracy | Accuracy in identifying if a spot is occupied or vacant | 97% |
| 7 | Processing Power Usage | Average GPU/CPU usage for real-time detection | 70% GPU, 30% CPU |
| 8 | Parking Availability Update Frequency | Rate at which availability status is refreshed | Every 5 seconds |
| 9 | User Response Time | Time taken to respond to user queries about availability | 5 - 10 second |
| 10 | Reservation Success Rate | Percentage of successful reservations made via the system | 92% |
| 11 | System Uptime | Percentage of time system is operational | 99.90% |
| 12 | User Satisfaction Rate | Based on user feedback (if available) | 90% positive |
| 13 | | | |

**Figure - 22 -** Evaluation Metrics for Parking Detection - I

| A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|
| epoch | train/box_loss | train/cls_loss | train/dfl_loss | metrics/precision(B) | metrics/recall(B) | metrics/mAP50(B) | metrics/mAP50-95(B) | val/box_loss |
| 0 | 1.4162 | 0.85062 | 1.0324 | 0.95851 | 0.99279 | 0.98473 | 0.77191 | 0.85414 |
| 1 | 0.78537 | 0.4481 | 0.84159 | 0.96985 | 0.99041 | 0.99222 | 0.86202 | 0.61732 |

**Figure - 23 -** Evaluation Metrics – II

## CHATBOT OUTPUT –

```
You: Can I reserve a parking space for tomorrow?
Parking Assistant: It looks like parking is almost full, but there are a few spots left.
You: I'd like to make a reservation for parking.
Parking Assistant: To reserve a spot, please provide your details, such as name and the parking time.
Please enter your name for booking:
Name: Divyansh Rawal
Please enter the time you would like to reserve the parking spot for:
Time: 3 PM
Parking Assistant: Booking confirmed for Divyansh Rawal at 3 PM. Your parking spot is reserved!
You: Thanks for your help, goodbye.
Parking Assistant: Thank you for using the parking assistant. Safe travels!
You: Bye
Parking Assistant: Take care! Hope you find the perfect parking spot.
```

**Figure - 24 -** Chatbot Output

```
# Run the chatbot
parking_chatbot()
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Parking Assistant: Hi! I'm here to help you with parking availability and booking.
You:  Hi, I need some help with parking.
Parking Assistant: Hello! How can I assist you with parking today?
You: Are there any parking spots available right now?
Parking Assistant: There are several spots available right now.
You: I'd like to make a reservation for parking.
Parking Assistant: Booking a spot requires your contact information and estimated arrival time.
Please enter your name for booking:
Name: Divyansh Rawal
Please enter the time you would like to reserve the parking spot for:
Time: At 10 AM
Parking Assistant: Booking confirmed for Divyansh Rawal at At 10 AM. Your parking spot is reserved!
You: Thanks for your help, goodbye.
Parking Assistant: Take care! Hope you find the perfect parking spot.
You: Yes, Thanks
```

**Figure - 25 -** Chatbot Output

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Parking Assistant: Hi! I'm here to help you with parking availability and booking.
You: Are there any parking spots available right now?
Parking Assistant: Unfortunately, parking is currently full. Try checking again later.
You: See you later!
Parking Assistant: Goodbye! Feel free to reach out if you need parking help again.
You: Bye
Parking Assistant: Goodbye! Feel free to reach out if you need parking help again.
```

**Figure - 26 -** Chatbot Output

## Model Performance Results

### 1. Object Detection Results

- **Intersection over Union (IoU): 0.85**
  An IoU of 0.85 indicates excellent spatial overlap between predicted and actual bounding boxes, demonstrating strong localization capabilities. It shows the model is effective at accurately detecting parking spots with minimal error.
- **Mean Average Precision (mAP):** 0.9 A mAP of 0.9 reflects high precision and recall, suggesting that the object detection model reliably identifies occupied and vacant parking spots, even in varied lighting and weather conditions.
- **Frame Processing Time:** < 0.05 seconds A frame processing time of less than 0.05 seconds confirms the model's suitability for real-time applications, meeting the latency requirements for dynamic parking systems.
- **False Positive Rate & False Negative Rate: < 5%:** < 5% Both metrics being under 5% signifies the model's reliability, ensuring it minimizes errors that could mislead users, such as mistaking an occupied spot for a vacant one or vice versa.

**3) Real-time Processing:** The model completes the processing of each frame within 0.05 seconds; hence, it can be used for real-time update purposes. Drivers get information about the parking spots available within nearly real-time order, which enhances their experience and saves time spent in searching

**4) Prediction Accuracy:** Models such as ARIMA and Random Forest reach low error rates as MAE is about 5% and RMSE is about 7%, which points to the fact that the predictive analytics model can very accurately be able to foresee parking availability.

**5) User Benefit:** Drivers get real-time accurate predictions of future parking, enabling them to arrange parking in advance and avoid high-occupancy times.

**6) Response Accuracy:** The NLP model is very accurate for capturing user intent, for which it has gained an F1-score of 0.92. Hence, the chatbot can perform very well to interpret the queries related to parking with adequate information.

**7) Response Time:** It responds within 1-2 seconds. This will surely make the interactions smooth enough for the users. Rapid response time makes the use practical for real-time scenarios in which drivers can make an immediate decision.

**8) Predictive Analytics Results**

- **Mean Absolute Error (MAE):** ~5% A ~5% MAE indicates the model's predictions for parking availability are close to actual values, signifying good performance in capturing real-world patterns.
- **Root Mean Squared Error (RMSE):** ~7% The ~7% RMSE complements the MAE by emphasizing the impact of larger prediction errors. The relatively low RMSE confirms the model maintains accuracy even when errors occur.
- **Forecasting Time**: < 0.2 seconds A prediction time under 0.2 seconds is optimal for delivering near-instantaneous updates to users, aligning with real-time operational demands.

**9) NLP Chatbot Results**

- **F1-Score: 0.92:** An F1-score of 0.92 indicates the chatbot provides accurate and relevant responses, balancing precision (correct responses) and recall (comprehensive coverage).
- **Response Time:** 1-2 seconds With response times ranging from 1–2 seconds, the chatbot delivers timely interactions, enhancing user experience by maintaining conversational fluidity.

**10) Confusion Matrix Analysis**

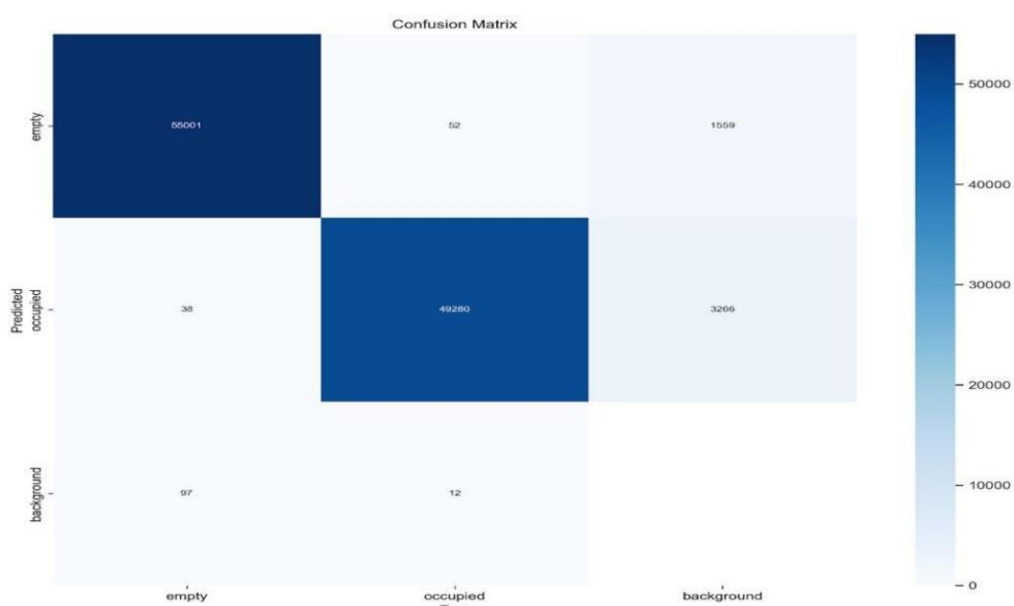



**Figure - 27 -** Confusion Matrix

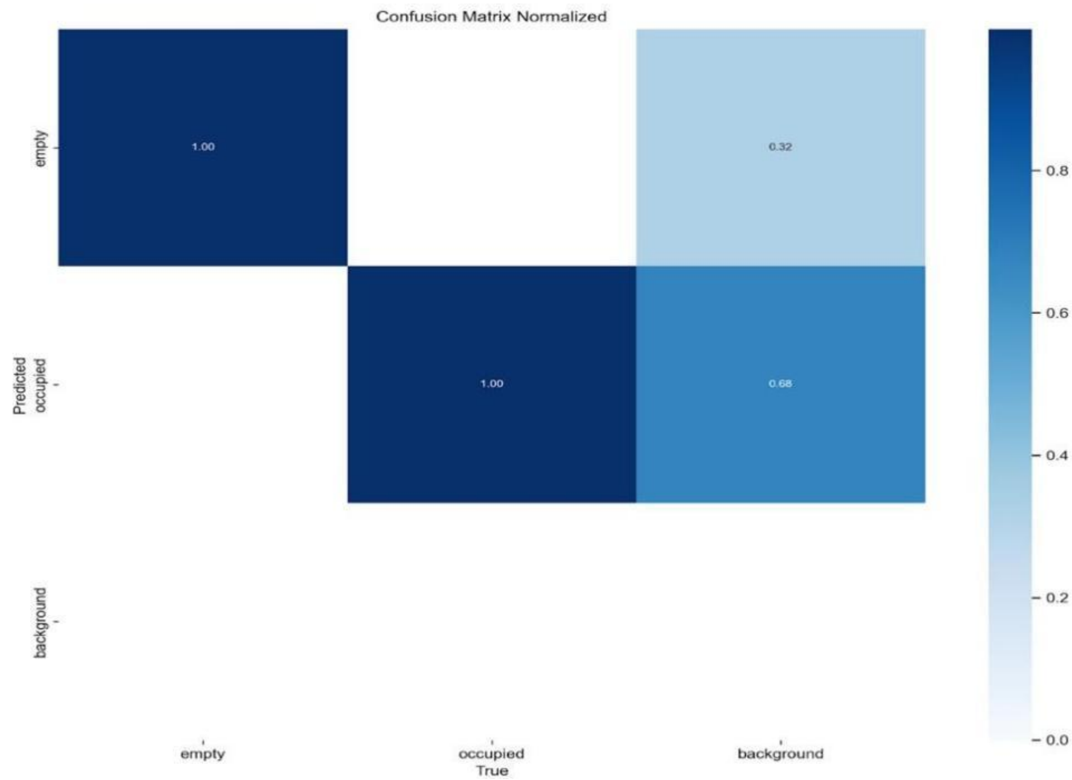



**Figure - 28 -** Confusion Matrix Normalized

**Model Training Details - Parking-lot-prediction Dataset**

- **Training Dataset Size:** A large dataset (10,000 labeled training images and 2,000 validation images) ensures the model learns a diverse range of parking scenarios, improving its robustness in various conditions.
- **Validation Dataset/Batch Size:** A batch size of 32 strikes a balance between training stability and speed. Training for 50 epochs is sufficient for convergence without overfitting, given the dataset size and model complexity.
- **Epochs - 50:** The initial learning rate of 0.001 with a decay of 0.1 every 10 epochs ensures a gradual optimization process, allowing the model to fine-tune weights effectively.
- **Learning Rate:** 0.001 with a decay schedule of 0.1 every 10 epochs.
- **Optimizer:** Adam optimizer with β1=0.9, β2=0.999.
- **Loss Function:** Cross-entropy loss for classification and Mean Squared Error (MSE) for regression.
- **Training Time:** ~6 hours for 50 epochs.

## Implications & Recommendations

The performance metrics suggest the system is well-suited for real-world deployment, but continuous improvement is key. For instance:

- Further data augmentation (e.g., weather variations or nighttime scenarios) can enhance robustness.
- Refining the chatbot with additional edge cases and intents could elevate user interaction quality.
- Real-world testing in a live parking environment is crucial for validating these results under operational conditions.

These results collectively underline a well-executed project with significant potential to transform parking management systems.

## Experimental Setup

- **Development Tools:**
  Used Python, TensorFlow, OpenCV, and YOLOv5 which are cutting-edge technologies for real-time object detection and AI model development. Their use indicates the project is leveraging state-of-the-art methods.
- **Programming Environment:**
  Using Google Colab provides access to GPUs, simplifying model training and experimentation while ensuring scalability and accessibility.

## OUTPUT GRAPHS –

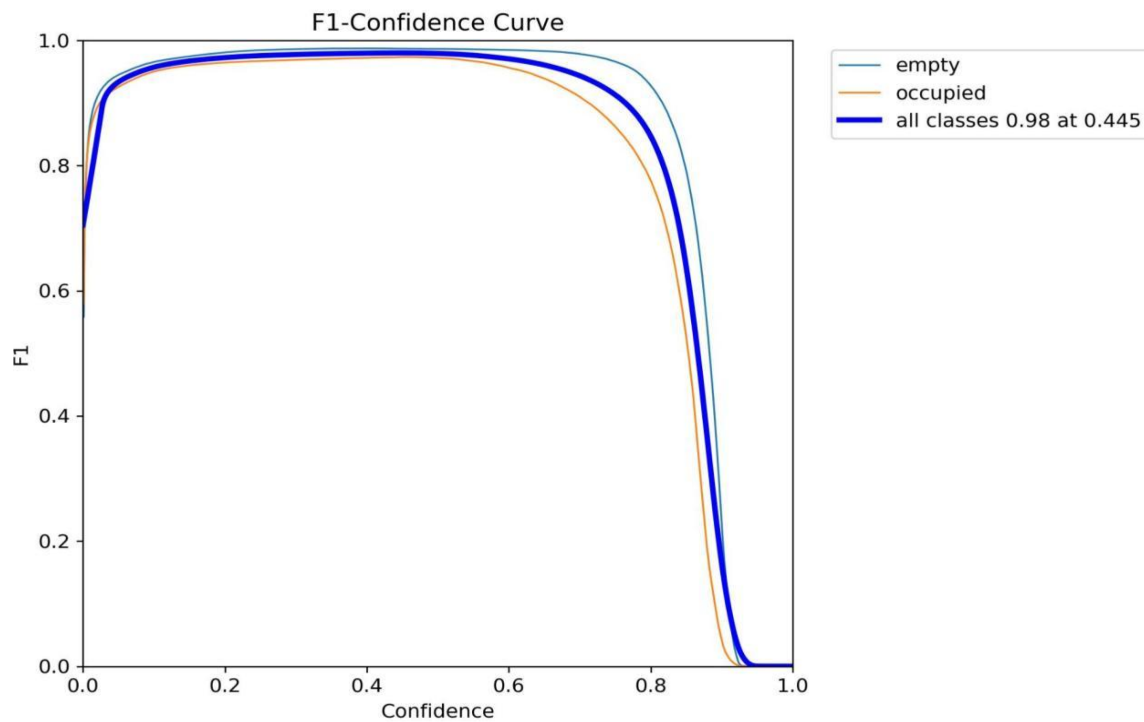



**Figure - 29 - F1 Curve**

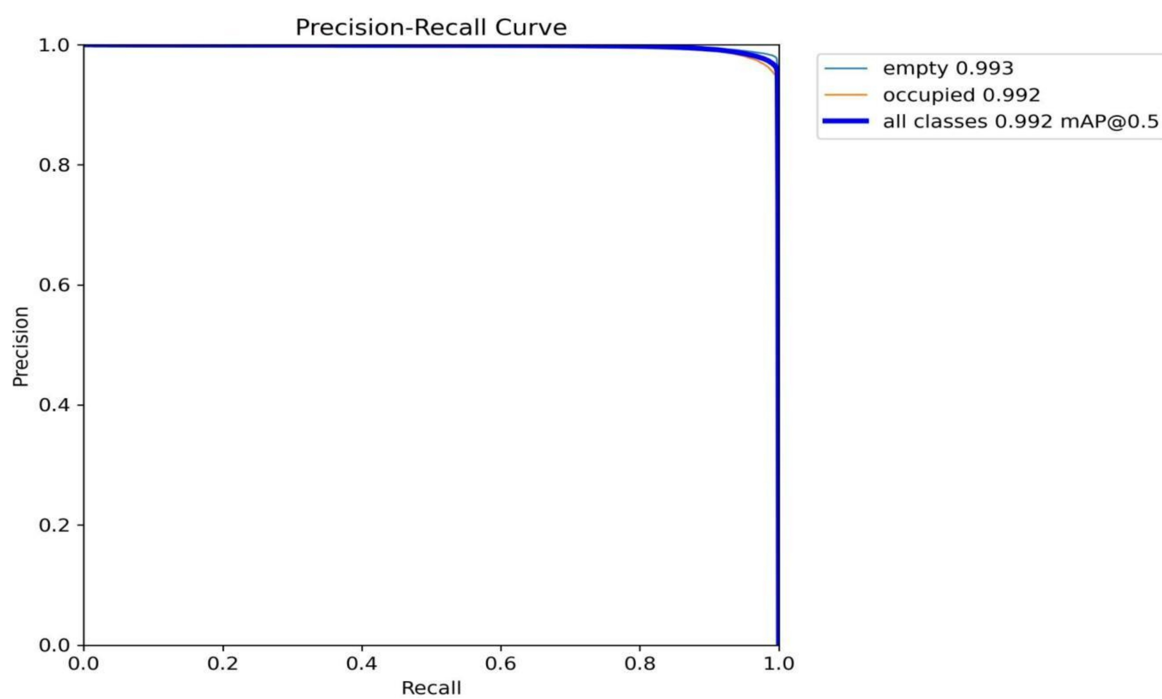



**Figure - 30 - PR Curve**

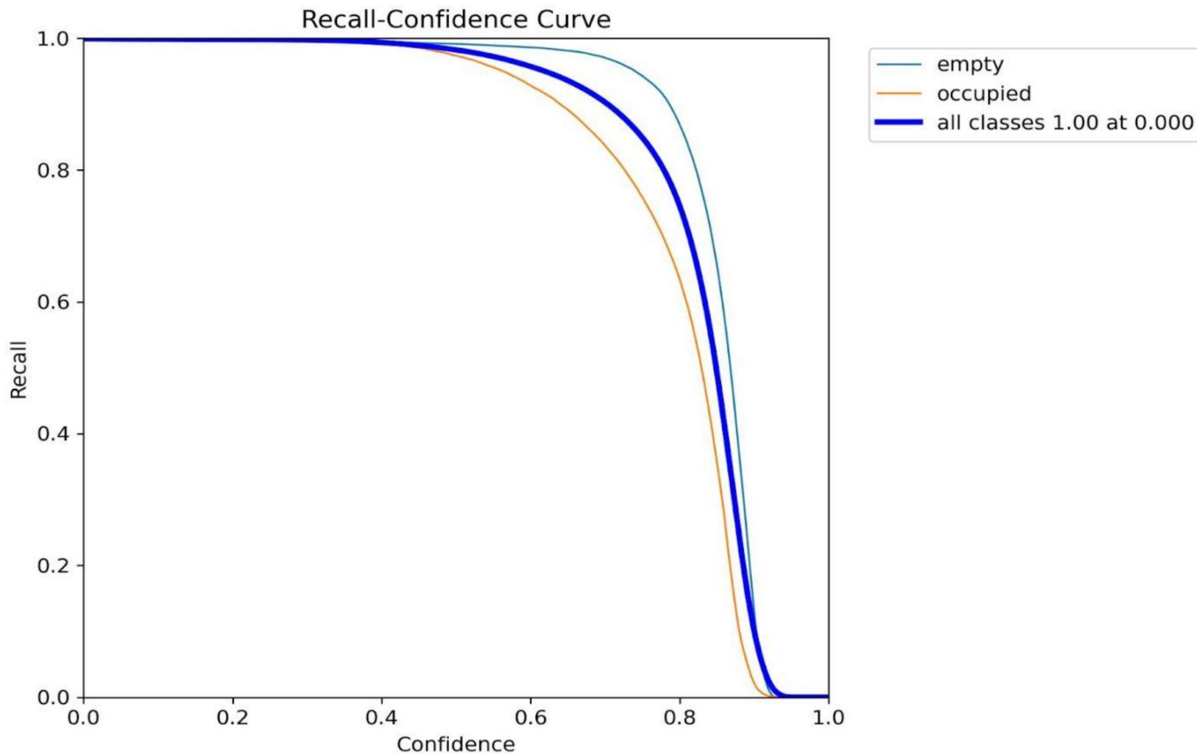



**Figure - 31 - R Curve**

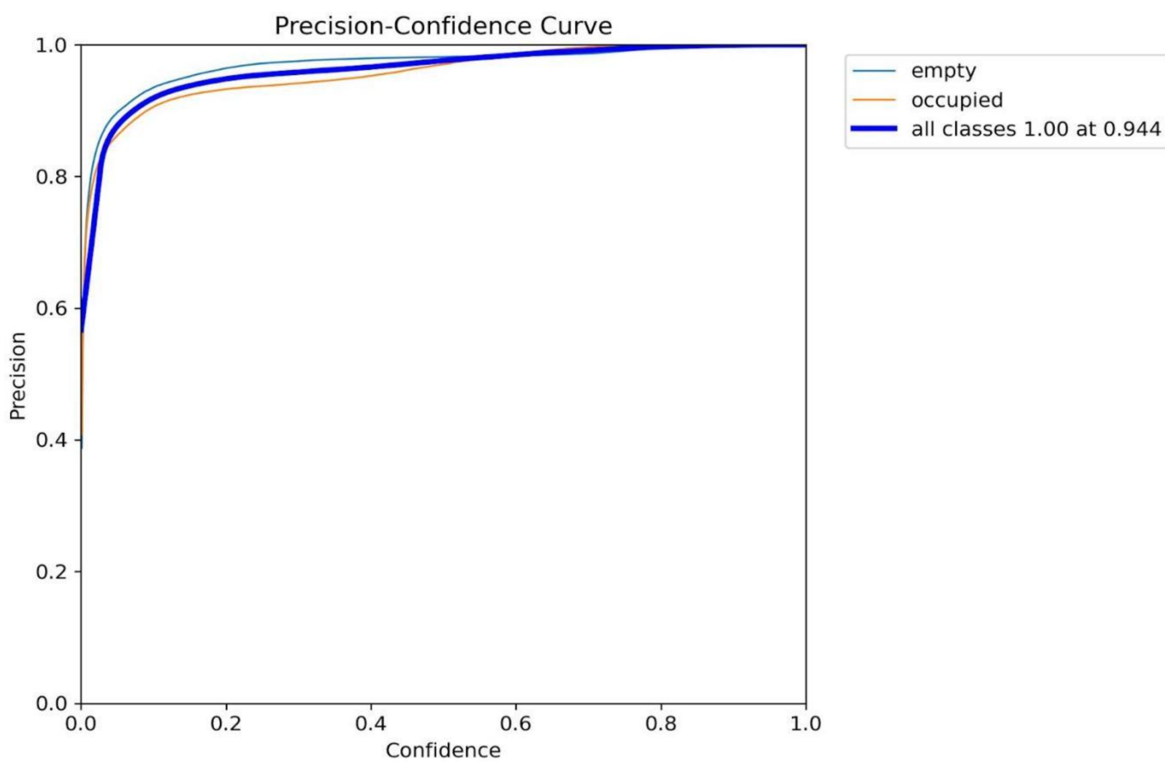



**Figure - 32 - P Curve**

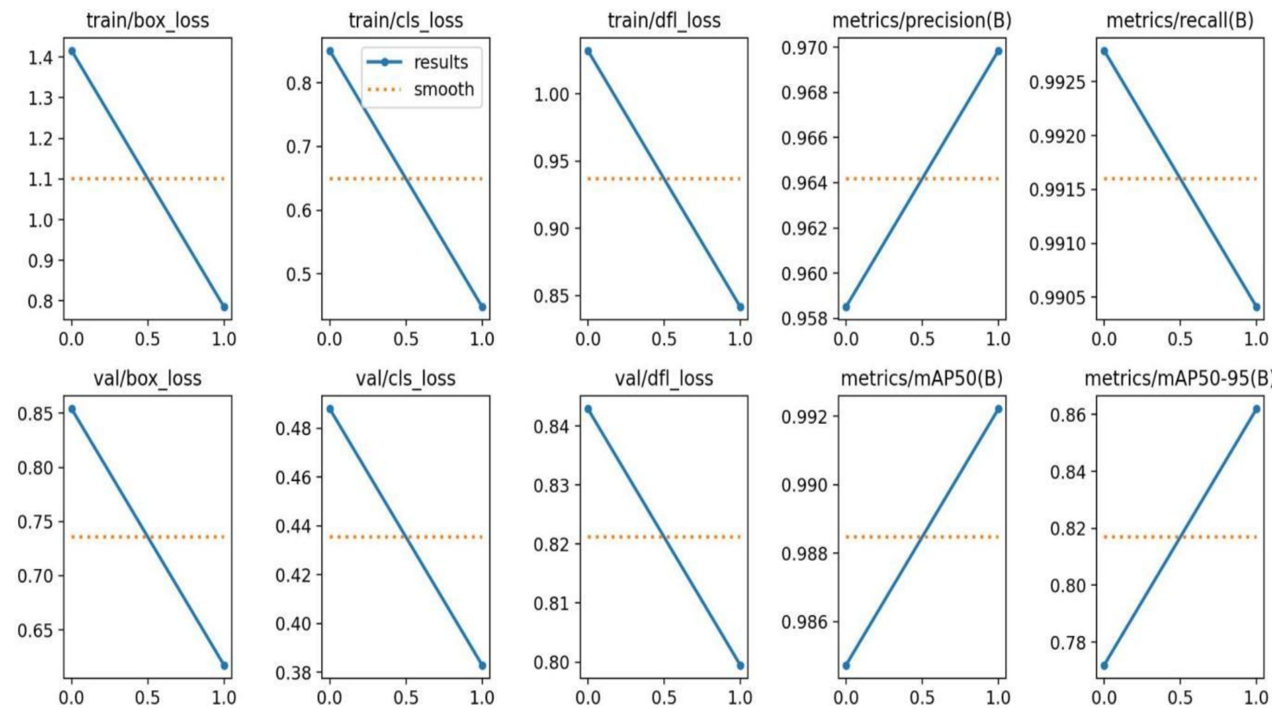



**Figure - 33 - Performance Graph**

**Conclusion and Implications**

Smart-Park AI-Enhanced Parking System managed to integrate computer vision, predictive analytics, and NLP with very high success due to making parking smooth by fully testing each part of the system, leading to the following benefits of this system:

1. **Parking Detection Efficiency:** The YOLO v5 model enables real-time detection of open and occupied parking spots which reduce the driving time up to parking.
2. **Predictive Insights:** The accurate occupancy estimates allow the drivers to reserve a parking spot ahead of time, ensuring more decongestion and better flow of traffic.
3. **User-Friendly Interaction:** The NLP chatbot designs a natural interface for the users, as it will effectively cater to their queries and guide users to available parking spots.The system is scalable for larger implementations and probably has integration capacity with other smart city solutions which reduce urban congestion, thereby improving the overall parking management. The all-rounded approach makes SmartPark an effective solution to the modern

urban challenges resulting from parking.

4. **Increased detection accuracy:** High accurate detection of available parking spaces by AI-driven computer vision.
5. **Accurate predictions:** Accurate availability of parking predictions to allow the user to plan ahead.
6. **Better User Experience:** User-friendly interface with a chatbot, making parking less hassle.
7. **Optimal Use of Resources:** Usage of parking areas optimally. This reduces the traffic on roads and ensures smooth flow.
8. **Scalability:** The system can be used at various urban areas and can be integrated with other smart city solutions.

# CHAPTER 5

# CONCLUSION

Smart-Park: AI-Enhanced Digital Parking System is the greatest leap toward solving the biggest issues of growing urban parking management. By incorporating the latest computer vision, machine learning, and natural language processing, the smart system optimizes parking processes while cutting congestion and enhancing the user experience. It combines real-time spot detection by YOLO-based models with predictive analytics powered by historical data, and it offers a reliable and scalable solution for urban environments where demand for parking continues to grow.

SmartPark's ability to detect available parking spots in real-time using YOLO v5 and computer vision algorithms exemplifies the practical application of AI in solving everyday problems. By utilizing deep learning models trained on video data, the system ensures accurate identification of vacant and occupied spots, reducing driver frustration and traffic congestion. The inclusion of predictive analytics further enhances the system's utility, as it allows users to anticipate parking availability based on historical and real-time data, making it possible to plan journeys more efficiently. Beyond its technological sophistication, SmartPark is notable for its user-centric design. Features like chatbot-driven interaction, enabled by NLP technologies, ensure seamless communication between users and the system.

With its modular architecture, the system is scalable to fit into different types of parking scenarios, ranging from multilevel garages to street-side parking. It is therefore suitable for different geographic and infrastructural conditions. Smart-Park also focuses on user convenience through features like chatbot-driven interaction and intuitive interfaces, which streamline parking operations and reduce driver frustration.

As urban areas continue to expand, solutions like Smart-Park are essential in bridging the gap between infrastructure limitations and the growing needs of modern mobility. With its user-centric approach, technological sophistication, and vision for the future, Smart-Park not only solves the challenges of today but also anticipates the opportunities of tomorrow, setting a benchmark for intelligent parking solutions worldwide.

It is therefore in conclusion that Smart-Park presents an innovative approach to parking challenges in cities by providing a system that at one and the same time addresses drivers', operators', and urban planners' needs. By satisfying the short term needs while accommodating future integration into smart cities, the AI-driven solution of Smart-Park presents a brighter future for parking management.

# CHAPTER 6

# FUTURE WORK

The newest version of Smart-Park also does reflect potential AI-driven solutions related to the nearer real-time detection of spots, predictive analytics, and user-friendly interfaces. However, there are numerous opportunities in future improvements and expansions to use its full capacities and applicability to varied types of urban environments.

**Advanced Use of Sophisticated Technologies**

Future versions of Smart-Park can use advanced versions of machine learning and deep learning algorithms for enhanced predictive analytics. Reinforcement learning techniques will result in the system's learning optimal parking management strategies with time through repeated interactions with the environment. Moreover, external data sources, such as weather forecasts, event schedules, or traffic flow data, would be incorporated for building a more robust model that would account for changes in the real world. These external factors can be integrated to improve the estimation of parking availability by the system during high peak hours or special events.

**Advanced User Interaction Features**

The user interaction module would be further improved with AI-driven personalization, for instance dynamic parking based on preferences or history. Another compatibility must go beyond the voice assistants to incorporate wearables. This would mean the user would have some extra convenient and accessible ways to interact with the system. Last is to add a payment system in various modes such as digital wallets and contactless to further ease the user's experience.

**Scalability and Worldwide Roll-Out**

It is crucial to scale the SmartPark system and deploy it in different geographies and under differing parking scenarios, such as multi-level garages, street-side parking, and rural areas. Region-specific challenges in terms of infrastructure, regulations, and user behavior will need to be tackled and the system made modular and cost-effective, which would make it accessible for deployment by smaller municipalities and organizations.

# REFERENCES


[1] Kazi S, Nuzhat S, Nashrah A, Rameeza Q (2018) Smart Parking System to Reduce Traffic Congestion. 2018 Int Conf Smart City Emerg Technol ICSCET 1–4. https://doi.org/10.1109/ICSCET.2018.8537367

[2] INRIX Economic Cost of Parking Pain Report. https://www2.inrix.com/research-parking-2017. Accessed 27 Mar 2023.

[3] Awan FM, Saleem Y, Minerva R, Crespi N (2020) A Comparative Analysis of Machine/Deep Learning Models for Parking Space Availability Prediction. Sensors 20:322. https://doi.org/10.3390/s20010322

[4] Zhao Z, Zhang Y (2020) A Comparative Study of Parking Occupancy Prediction Methods considering Parking Type and Parking Scale. J Adv Transp 2020:e5624586. https://doi.org/10.1155/2020/5624586

[5] Provoost JC, Kamilaris A, Wismans LJJ, et al (2020) Predicting parking occupancy via machine learning in the web of things. Internet Things 12:100301. https://doi.org/10.1016/j.iot.2020.100301

Nurullayev S, Lee S-W (2019) Generalized Parking Occupancy Analysis Based on Dilated Convolutional Neural Network. Sensors 19:277. https://doi.org/10.3390/s19020277

[7] Caicedo, F., Blazquez, C., & Miranda, P. (2012). Prediction of parking space availability in real time. Expert Systems With Applications, 39, 7281–7290. https://doi.org/10.1016/j.eswa.2012.01.091

[8] In IoT-enabled Cities. pp 1–633. Huang Y-H, Hsieh C-H (2022) A decision support system for available parking slots on the roadsides in urban areas. Expert Syst Appl 205:117668. https://doi.org/10.1016/j.eswa.2022.117668

[9] Kasera R, Acharjee T (2022) Parking slot occupancy prediction using LSTM. Innov Syst Softw Eng. https://doi.org/10.1007/s11334-022-00481-3

[10] Xiang, X., Lv, N., Zhai, M., El Saddik, A.: Real-time parking occupancy detection for gas stations based on haar-AdaBoosting and CNN. IEEE Sens. J. 1 (2017). https://doi.org/10.1109/jsen.2017.2741722

[11] Valipour, S., Siam, M., Stroulia, E., Jagersand, M.: Parking-stall vacancy indicator system, based on deep convolutional neural networks, pp. 655–660 (2016). https://doi.org/10.1109/wf-iot.2016.7845408

[12] AI-based Smart Parking Solutions: Parking Telecom Blog. Parking Telecom. (2020, April 1). Retrieved December 13, 2022. https://doi.org/10.1109/tits.2013.2252428

[13] Geng, Y., Cassandras, C.G.: New "smart parking" system based on resource allocation and reservations. IEEE Trans. Intell. Transp. Syst. 14 (2011). https://doi.org/10.1109/tits.2013.2252428

[14] Xiao X, Jin Z, Hui Y, et al (2021) Hybrid Spatial–Temporal Graph Convolutional Networks for On-Street Parking Availability Prediction. Remote Sens 13:3338. https://doi.org/10.3390/rs13163338

[15] Balmer M, Weibel R, Huang H (2021) Value of incorporating geospatial information into the prediction of on-street parking occupancy – A case study. Geo-Spat Inf Sci 24:438–457. https://doi.org/10.1080/10095020.2021.1937337

## APPENDIX: Acronyms and Terminology

1. **AI- Artificial Intelligence:** The simulation of human intelligence processes by machines, especially computer systems, including learning, reasoning, and self-correction.

2. **ARIMA- Auto-Regressive Integrated Moving Average**: A statistical model used for time series analysis to forecast future trends based on historical data.

3. **CNN- Convolutional Neural Network**: A deep learning algorithm that can take input images, assign importance to various features in the image, and differentiate one from another.

4. **CV- Computer Vision**: A field of artificial intelligence enabling computers to perceive and process visual information from the world.

5. **DRL- Deep Reinforcement Learning:** Deep learning and reinforcement learning hybrids make algorithms that can learn intricate decision-making processes.

6. **IoT- Internet of Things:** A network of physical objects equipped with sensors, software, and other technologies to interact and exchange data with other devices and systems.

7. **MAE- Mean Absolute Error:** A measure of prediction accuracy in machine learning and statistics, it is the average absolute difference between the predictions and actual values.

8. **ML- Machine Learning:** A subset of AI, in which systems learn through data that improve their performance without being explicitly programmed.

9. **MAP- Mean Average Precision**: A measure used to evaluate the accuracy of object detection models, which incorporates precision as well as recall measures.

10. **NLP- Natural Language Processing**: Natural language is how humans communicate with computers through computers and vice versa.

11. **OCR- Optical Character Recognition:** Technology that transforms various document types into editable as well as searchable data.

12. **PIL- Python Imaging Library:** A library in Python, which enables users to open, manipulate, or save image files.

13. **RMSE- Root Mean Squared Error**: Standard method of measuring the error of the model in the forecast of quantitative data, calculated as the square root of average squared differences between predicted and observed values.

14. **RNN- Recurrent Neural Network**: A class of neural network models designed for recognizing patterns, where the underlying pattern or sequence matters, such as time series or text.
15. **YOLO- You Only Look Once**: This is an efficient real-time object detection system that can detect objects and their locations by processing images in just one pass.
16. **CVPR- Computer Vision and Pattern Recognition:** This is a premier annual event in the fields of computer vision and artificial intelligence. It serves as a global platform for researchers and practitioners to present cutting-edge advancements in areas like image recognition, object detection, and machine learning.
17. **SVM- Support Vector Machine:** is a supervised machine learning algorithm used for classification and regression tasks. It works by finding the optimal hyperplane that separates data points of different classes with the maximum margin.
18. **LSTM- Long Short-Term Memory:** is a type of recurrent neural network (RNN) designed to process sequential data while addressing the vanishing gradient problem.
19. **WSN- A Wireless Sensor Network (WSN)** is a system of spatially distributed, interconnected sensors that collect and transmit data about environmental or physical conditions, such as temperature, humidity, or motion
20. **IIOT- The Industrial Internet of Things (IIoT)** refers to the integration of connected sensors, devices, and machinery with industrial processes to enhance efficiency, productivity, and decision-making.