Parking Pixels: Python's Vision for Urban Space Optimization

Sakthivel S, Dr. V. Dhanakoti, Vedavarshini. K, Sundaresan. C
Department of Computer Science Engineering

Abstract - As urbanization accelerates and vehicle numbers surge, the demand for effective parking management systems grows. This study introduces the creation of a Python-based parking space detection application. The application's objective is to furnish instantaneous data on vacant parking spots within assigned parking zones, thereby refining parking experiences and alleviating congestion. Central elements encompass a user-centric interface facilitating smooth interaction across diverse devices, graphical depictions of parking lot layouts, and live updates regarding space availability. The core functionality lies in pinpointing available parking spots using image-processing techniques and computer vision algorithms. Camera integration and video footage analysis identify vacant spaces based on predefined criteria. Robust backend logic maintains a dynamic database of parking space statuses, reconciling conflicting inputs from multiple sources. The application can be packaged for distribution on various platforms, facilitating accessibility. Continuous improvement and user feedback mechanisms drive iterative enhancements, ensuring the application's functionality evolves over time.

Keywords--(Python Kivy, YOLOv9, Object detection, Computer vision, Surveillance, User interface, Real-time processing, Parking management, Data visualization)

I. INTRODUCTION

In today's fast-paced urban landscapes, finding a parking space is often a time-consuming and frustrating experience for drivers. The escalating number of vehicles on the road coupled with limited parking infrastructure has exacerbated this challenge, leading to congestion, increased fuel consumption, and heightened stress levels among drivers. To address this pervasive issue, we propose the development of an innovative mobile application that harnesses the power of computer vision and deep learning technology to detect and present up-to-the-minute updates on parking space availability.

This project addresses the shortcomings of traditional parking management systems. Traditional methods rely on manual checks or static sensors, which can be time-consuming, error-prone, and inefficient in dynamically changing parking environments. Integrating computer vision into a mobile app creates a dynamic solution for real-time parking space detection.

The key to a smoother parking experience lies in efficient space detection and management. In busy urban environments, finding available parking spots can be challenging, leading to congestion, wasted time, and increased pollution due to unnecessary circling by drivers. The scope of this project encompasses developing a user-friendly mobile app that uses computer vision algorithms to identify and showcase unoccupied parking spaces instantly, providing drivers with timely and precise details.

Our solution seeks to revolutionize this process by leveraging advanced technologies to provide drivers with accurate and up-to-date information on parking availability. By integrating computer vision capabilities with YOLO, an efficient object detection algorithm, into a mobile application built with Python Kivy, we can empower drivers to locate vacant parking spaces swiftly and conveniently.

YOLOv9 stands out as a cutting-edge object detection model renowned for its rapidity, precision, and effectiveness. It builds upon the YOLOv8 architecture. Innovations in YOLOv9 Programmable Gradient Information (PGI): This technique addresses the information bottleneck issue that can occur in deep neural networks. Generalized Efficient Layer Aggregation Network (GELAN): This is a new network architecture designed to be lightweight and efficient.

This research is significant as it aims to bridge the gap between emerging technologies and practical solutions for everyday challenges. By developing a mobile app that leverages Python with Kivy for parking space detection, this

study seeks to demonstrate the feasibility and benefits of using computer vision in enhancing urban mobility and parking efficiency.

II. Related Work

A. Duy-Linh Nguyen et al [1] introduced "YOLO5PKLot," a cutting-edge network architecture crafted explicitly for real-time parking space detection in smart parking management systems. It addresses the growing need for efficient parking solutions amid increasing vehicle numbers worldwide. Traditional sensor-based methods are costly, prompting a shift towards vision-based approaches leveraging deep learning, with YOLOv5 serving as the foundation. In methodological enhancements, YOLO5PKLot optimizes the YOLOv5 architecture by integrating lightweight Ghost Bottleneck and Spatial Pyramid Pooling components into the backbone. This reduces network parameters and computational complexity while enhancing detection performance. Furthermore, modifications to the detection head and anchor resizing contribute to improved accuracy in parking spacedetection. Experimental validation involves training and evaluating YOLO5PKLot on the Parking Lot Dataset, comprising diverse parking lot images. The network actively trains on a GPU using PyTorch, incorporating data augmentation techniques to significantly boost its performance. YOLO5PKLot achieves impressive results, reaching a mAP of 99.6% on the evaluation set, demonstrating its success in parking lot detection. Comparative analysis reveals superior performance over previous networks, with fewer parameters and lower computational demands. YOLO5PKLot exhibits real-time inference capability, making it suitable for deployment in practical parking management systems. The paper includes ablation studies to assess variations in backbone design and head numbers, offering insights into optimization strategies. In conclusion, it highlights YOLO5PKLot's efficiency, effectiveness, and applicability in smart parking systems. Future research may explore integrating attention modules to address specific detection challenges, further enhancing network performance.

B. Lun-Chi Chen et al [2] The study focuses on smart city infrastructure, the 'Video-Based Parking Occupancy Detection for Smart Control System' study explores a cost-effective, weather-resistant method for outdoor parking detection using existing streetlights. This is vital for reliable smart surveillance. The research proposes a new, Jetson TX2-powered method replacing expensive, sensor-based systems. It leverages YOLO v3 and MobileNet v2 for accurate street parking occupancy identification. Functioning as an area-based system with voting, it ensures reliable occupancy recognition. Additionally, the system controls streetlights, maintaining half-brightness at night and increasing it upon vehicle detection, promoting energy conservation alongside parking detection. Testing with the CNRPark+ EXT dataset, simulations, and real-world camera footage confirmed the framework's ability to achieve stable parking occupancy detection on streets, benefiting urban planning and reducing social costs of city infrastructure development.

C. Abrar Fahim et al [3] The journal article delves into the realm of smart parking systems (SPS) within the broader context of smart cities and the Internet of Things (IoT). It outlines the transformative impact of IoT on modern life, emphasizing its role in connecting devices and enhancing urban services. With a focus on addressing parking challenges in densely populated urban areas, the article highlights the urgency of efficient parking solutions to combat congestion, fuel wastage, and pollution. A meticulous literature review methodology is presented, detailing stages of planning, review, and results. The planning stage sets research objectives and formulates key questions, guiding the search process. The review stage involves a comprehensive search across reputable scientific databases, resulting in the selection of relevant literature. Finally, the result stage synthesizes findings to identify technological trends in SPS development. The article offers a detailed classification of SPS based on criteria such as approaches/methods, sensors, networking technologies, computational approaches, and services provided. It categorizes SPS approaches into Wireless Sensor Networks, Multi-Agent Systems, and Computer Vision/Image Processing, among others. Similarly, sensors used in SPS are classified as Infrared Rays, Cellular Sensors, and Camera-based systems. Furthermore, the article provides insights into the advantages and disadvantages of existing SPS solutions, aiming to assist researchers, system designers, and policymakers in making informed decisions. By offering a systematic comparison of SPS approaches and highlighting their strengths and limitations, the article contributes to a deeper understanding of SPS technology and its implications for urban mobility and sustainability.

D. Amisha Narkhede et al [4] The project begins with modeling and drafting components using software like SolidWorks. This step allows for a detailed visualization of the system's layout and spatial arrangement based on selected dimensions. Subsequently, a schematic of the smart car parking system is designed using software

such as Eagle. This schematic acts as a roadmap, detailing the electrical connections and interactions between circuit components. It outlines how the various parts of the system will be arranged and connected to ensure proper functioning. A significant challenge addressed in the methodology is the inconsistency of infrared (IR) rays used for vehicle detection. To mitigate this challenge, an IR emitter is employed to project radiation light, which is then received by an IR receiver. The receiver translates the received radiation into an electrical signal, indicating the presence of a vehicle. Moreover, adjustments are made to ensure that the voltage of the circuit increases as the distance between the emitter and receiver decreases, signaling the proximity of a vehicle accurately. After design and schematics, prototypes undergo rigorous testing in diverse conditions to ensure the smart parking system functions reliably. This testing phase enables the identification of any potential issues or areas for improvement, ensuring that the final system meets the desired performance standards. Overall, the methodology for developing smart car parking systems involves a systematic approach encompassing modeling, schematic design, addressing technical challenges, and prototype testing to deliver a robust and efficient parking solution.

E. Aashish Joshi et al [5] The project introduces a Parking Management System (PMS) designed to efficiently manage parking slots within a parking structure. It offers a simplified solution for vendors and parking lot operators, facilitating easy entry and exit of vehicles while maintaining an up-to-date database. Targeting parking optimization, congestion reduction, pollution minimization, user experience improvement, and revenue generation, this project uses Arduino as the central controller. IR sensors at entry/exit and parking spaces detect vehicles and manage slot availability, while a servo motor operates the gate and an LCD displays remaining spots. Project resources include a high-speed internet connection, Arduino Uno IDE, power supply, and an android device. The system accommodates four parking slots, with IR sensors strategically positioned for vehicle detection. Results include quick, automated parking, secure vehicle retrieval, space-efficient parking, and low maintenance. The system is cost-effective, highly automated, and user-friendly, requiring minimal manual intervention. It ensures safety through vehicle reorientation and offers real-time data insights for parking lot owners. Concisely, this project offers a complete picture of the Parking Management System, showcasing its effectiveness in tackling parking issues and enhancing the overall experience.

III. APPROACH

A. *User Input:*
- Pre-recorded videos: Users can select a video file stored on their device or from a designated location.
- Live video streams: Real-time parking lot views are accessible through surveillance cameras. This can be achieved by connecting to the camera's IP address or using Kivy's camera access functionalities to stream video directly within the application.

B. *Region Selection:*
- Users can define parking regions by interacting with the video feed displayed on the application interface.
- Drawing bounding boxes or polygons: Users can use touch events or mouse clicks to draw shapes directly onto the video frame, outlining the boundaries of parking spaces.
- Editing and refining: Users should have the option to edit or delete existing regions and add new ones as needed. This ensures flexibility and accuracy in defining parking areas.

C. *Region Saving:*
- Once the user has defined parking regions, the application should save this information for future use.
- Saving coordinates or dimensions: Each defined region should be associated with specific coordinates or dimensions within the video frame. This information can be stored in a configuration file or database, allowing the application to retrieve it when needed.

D. *YOLOv9 Integration:*
- YOLOv9 (You Only Look Once version 9) is a state-of-the-art object detection model, performing real-time identification and categorization of objects. Integrating YOLOv9 into the application necessitates embedding its detection algorithm within the codebase.
- Configuration of YOLOv9 focuses on identifying vehicles within specified parking regions, discerning between occupied and unoccupied parking spaces by detecting vehicle presence within those areas.

E. *Vacant Parking Detection:*

- Following the processing of the video feed by YOLOv9, the application conducts an assessment to ascertain the quantity of available parking spaces.
- This assessment entails tallying the identified vehicles within each designated parking area and contrasting it with the total number of parking slots allocated to that region.
- The app must provide real-time parking availability with a constantly updated user interface.

F. *User Notification:*

- Once vacant parking slots are identified, the application notifies the user about their availability.
- Notifications can take the form of alerts or visual indicators within the application interface.
- Users should be informed about the number and locations of vacant parking slots, empowering them to make well-informed choices regarding their vehicle parking locations.

IV. METHOD

A. *YOLOv9 :*

YOLOv9 detects vehicles within defined parking regions, distinguishing between occupied and vacant slots. Integrated into the application, YOLOv9 utilizes its deep learning architecture to analyze video frames, supplying instantaneous updates on parking space availability. This enables users to conveniently view the current parking status and identify available slots. Overall, YOLOv9 enhances your smart car parking application by automating vehicle detection and occupancy monitoring, streamlining the parking experience.

We leveraged a pre-trained YOLOv9 model to enhance our object detection accuracy. The model, initially trained on the COCO dataset, was fine-tuned on our specific dataset comprising 3,437 samples. This included 3,186 samples for training, 169 for testing, and 82 for validation, to adapt the model to the nuances of parking slot detection.

One of the standout features of YOLO models has always been their speed. YOLOv9c continues this tradition by maintaining real-time detection capabilities while pushing the boundaries of accuracy. The model is designed to run efficiently on various hardware, from high-end GPUs to more constrained edge devices.

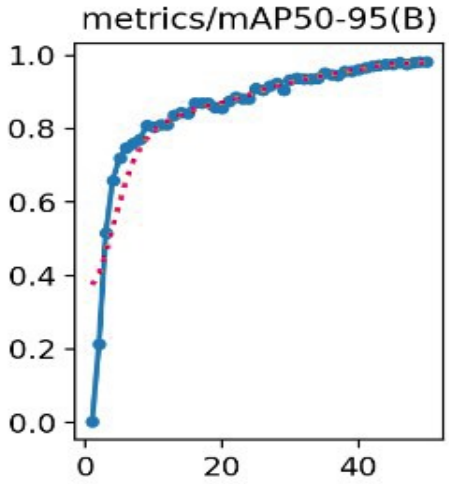

Fig. 1. Metrics/ mAP50-95 (mean Average Precision)

The provided graph illustrates the mAP50-95 (mean Average Precision) metric for a custom-trained YOLO v9 model over 50 training epochs. The x-axis represents the number of epochs, while the y-axis measures the mAP50-95, ranging from 0.0 to 1.0. The solid line with circular markers shows the mAP50-95 values for the user-trained model, initially rising steeply, indicating significant performance improvements in the early stages of training. As training continues, the rate of improvement slows, with the mAP50-95 values gradually approaching a plateau, suggesting the model's performance is stabilizing. Eventually, the curve flattens out near a value close to 1.0,

demonstrating that additional training epochs do not substantially enhance the model's performance further. This pattern signifies that the user-trained YOLO v9 model achieves high accuracy in object detection, reflected in the strong mAP50-95 performance metric.

TABLE I. BENCHMARK OF YOLOV9C(OUR) MODEL

Model	Size (MB)	Metrics /mAP50-95	Inference time (ms/im)	FPS
YOLOv9c-custom.pt	49.2	0.9839	33.35	30

The custom-trained YOLOv9c model, weighing 49.2 MB, achieves an impressive mAP of 0.9839, indicating high accuracy in object detection across various thresholds. It operates at an efficient speed of 33.35 milliseconds per image, translating to 30 frames per second (FPS), making it suitable for real-time applications like video surveillance and autonomous systems where both accuracy and speed are crucial.

B. *Kivy :*

Python Kivy offers an ideal framework for developing your smart car parking application using YOLOv9. Its cross-platform compatibility and extensive features enable efficient creation of user-friendly interfaces tailored to your project's needs. Seamlessly integrating with Python libraries allows for advanced data analysis and machine learning capabilities, enhancing the depth of your application. Additionally, Kivy's flexible deployment options ensure widespread accessibility across devices, maximizing your project's impact. Whether you're designing data visualization tools or interactive research platforms, Python Kivy drives innovation, elevating the user experience in your smart car parking application.

C. *Matplotlib:*

Matplotlib is utilized to create an interactive interface for users to mark parking regions on a video frame. This is achieved by leveraging Matplotlib's Polygon Selector, which allows users to draw bounding boxes or polygons directly on the video frame. The selected regions are then saved and processed for further analysis. Additionally, Matplotlib is employed to display the video feed along with the marked regions, providing users with real-time feedback and visual confirmation of their selections. Overall, Matplotlib serves a vital function in facilitating user interaction and improving the user experience of your smart car parking application.

D. *Opemcv:*

OpenCV, an open-source computer vision library, offers an extensive toolkit for real-time image and video processing. Written in C++, it boasts Python bindings, facilitating broad adoption across domains like robotics, augmented reality, and face recognition. With functions for resizing, filtering, and histogram equalization, it supports diverse image processing tasks. Key algorithms include feature detection, object detection, image segmentation, optical flow, and motion estimation. Integration with TensorFlow and PyTorch enables machine learning and deep learning tasks like model training and inference. OpenCV's versatility stems from its cross-platform compatibility and wide language support, making it ideal for developers on Windows, Linux, macOS, iOS, and Android.

E. *PyTouch*

PyTorch stands out as a potent framework for constructing and refining machine learning models, particularly deep neural networks. Its utilization of dynamic computation graphs sets it apart, as the graph is constructed dynamically during execution. These characteristic grants greater adaptability in model architecture and debugging. Renowned for its simplicity and Pythonic syntax, PyTorch proves to be more approachable and accessible for novices, facilitating a smoother learning curve. Due to its dynamic nature, debugging PyTorch models can be more straightforward as you can use standard Python debugging tools. PyTorch has a strong presence in the research community, with many cutting-edge research projects and libraries developed using PyTorch

V. SYSTEM ARCHITECTURE

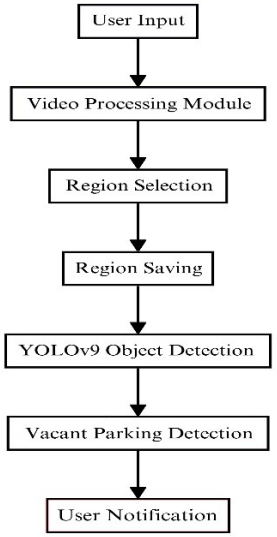

Fig. 2. Dataflow Diagram

- User Input: The user provides initial input by selecting a video file for processing.
- Video Processing Module: Processes the input video to extract frames and prepare the data for analysis.
- Region Selection: Specific regions of interest within the video frames (e.g., potential parking spots) are selected.
- Region Saving: Saves the selected regions for future reference during the object detection phase.
- YOLOv9 Object Detection: Uses the YOLOv9 model to detect and classify objects within the selected regions of the video frames.
- Vacant Parking Detection: Determines which parking spots are vacant by identifying the absence of cars in the selected regions.
- User Notification: Notifies the user about the availability of parking spots, typically through a visual or app notification.

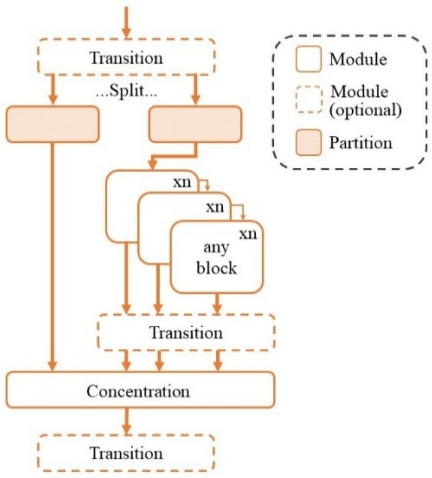

Fig. 3. GELAN architecture

The GELAN architecture depicted in the provided diagram represents a modular and flexible approach to neural network design. The architecture consists of several key components arranged in a structured sequence to enhance the model's learning capabilities.

- Transition Layers: These layers, found at the beginning and between other components, transform the data for subsequent processing.
- Splitting Mechanism: After an initial transition, the data is split into multiple parallel paths.
- Parallel Blocks: Each path processes the data through repeated sequences of operations (blocks), allowing for diverse feature extraction. The "any block" notation indicates flexibility in block types.
- Concatenation: The outputs from the parallel blocks are merged, combining the various features learned.
- Final Transition Layers: Further refine and prepare the data for the network's final stages.

This design allows GELAN to capture a wide range of features, improving model performance and adaptability.

VI. RESULT

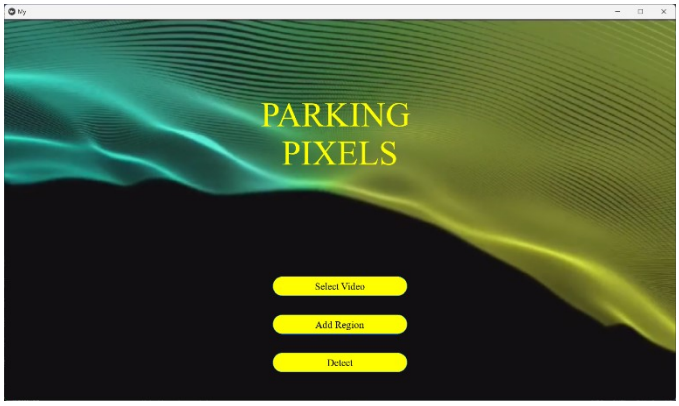

Fig. 4. Interface of Parking Pixels

The smart car parking application, developed using Python Kivy and YOLOv9, demonstrates efficient parking management and enhanced user experience. Leveraging an intuitive interface and advanced object detection algorithms, the application successfully addresses the challenges of parking slot management in urban environments.

Upon inputting pre-recorded videos or live streams, the system checks if a region file corresponding to the selected video exists. If it does not users can intuitively define parking regions within the surveillance area.

The system then processes this information, utilizing the custom trained YOLOv9 model to detect vehicle presence within the designated slots.

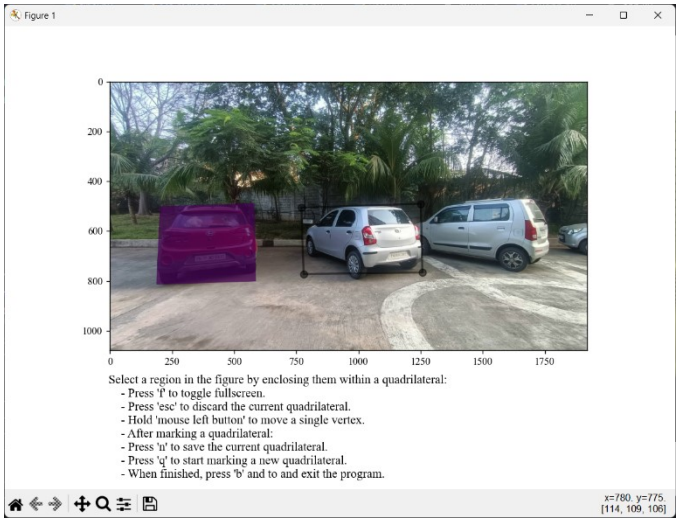

Fig. 5. Selecting region for car parking slot

The app interface displays real-time parking availability, allowing for informed decision-making and optimized parking resource utilization. The application's robust architecture enables seamless interaction between users and the parking management system, promoting ease of use and accessibility. By leveraging cutting-edge technologies, including Python Kivy for interface development and YOLOv9 for object detection, the project achieves accurate and efficient parking slot monitoring.

Fig. 6. Output Predicted Image with slot number

Overall, the implementation of the smart car parking application yields tangible benefits, including reduced congestion, minimized search time for parking, and improved overall traffic flow. This project showcases the transformative power of technology-driven solutions in tackling urban parking issues, laying the groundwork for the advancement of smarter and more sustainable urban landscapes.

VII. Future Work

Moving forward, future work for our project involves several key areas of development. We are initially focusing on incorporating cutting-edge machine learning methods like deep reinforcement to optimizing parking spot detection algorithms for better accuracy. Additionally, we will focus on continuously refining the user interface based on feedback to ensure a seamless experience across various platforms. Real-time optimization algorithms will be explored to dynamically adjust parking lot layouts, maximizing efficiency and minimizing congestion. We also plan to investigate opportunities for integration with broader smart city infrastructure, aiming to provide comprehensive mobility solutions. Further enhancements include multi-modal sensor fusion for improved accuracy and scalability, as well as strategies to enhance energy efficiency and sustainability. Collaboration with stakeholders will remain integral for gathering real-world data and informing future developments. By pursuing these endeavors, our project will further propel the evolution of intelligent parking management systems, thereby bolstering urban mobility and sustainability initiatives.

VIII. Conclusion

In summary, our parking space detection application's development encompasses several pivotal elements geared towards delivering users a streamlined and effective parking experience. Users have the flexibility to input either pre-recorded videos or access live surveillance footage from the parking lot. Through intuitive region selection functionalities, users can define parking regions by drawing bounding boxes or polygons directly onto the video feed, ensuring accuracy and flexibility. Once regions are defined, the application saves this information for future use, associating each region. By leveraging specific coordinates or dimensions within the video frame, our system integrates with YOLOv9, a cutting-edge object detection model, for instant recognition and classification of vehicles within designated parking areas. This seamless integration allows for precise identification of vacant parking spots in real-time. Our application continually monitors YOLOv9's output, promptly updating the user interface with alerts or visual cues to inform users of parking availability. By incorporating these functionalities, our

application aims to optimize the parking experience, reduce congestion, and improve overall efficiency in parking management systems.

REFERENCES

[1] Geethapriya. S, N. Duraimurugan, S.P. Chokkalingam, Real-Time Object Detection with Yolo, International Journal of Engineering and Advanced Technology (IJEAT), ISSN:2249-8958 (Online), Vol. 8, Issue.3S, PP.578-581, Feb 2019, DOI: https://doi.org/10.35940/ijeat.2249-8958.

[2] Yong-Hwan Lee, Hyochang Ahn, Vehicle Classification and Tracking Based on Deep Learning, Journal of Web Engineering, ISSN: 1544-5976 (Online Version), Vol.21, Issue.4, PP.1283–1294, Apr 2022, https://doi.org/10.13052/jwe1540-9589.21412.

[3] Deepak Mane, Prashant Kumbharkar, Sunil Sangve, Nirupama Earan, Komal Patil6Sakshi Bonde, A Metaphor Analysis on Vehicle License Plate Detection using Yolo-NAS and Yolov8, Journal of Electrical Systems, ISSN:1112-5209, Vol.20, Issue.1s, PP.152-164, Jan 2024, DOI: https://doi.org/10.52783/jes.761.

[4] Minar Mahmud Rafi, Siddharth Chakma, Asif Mahmud, Raj Xavier Rozario, Rukon Uddin Munna, Md. Abrar Abedin Wohra, Rakibul Haque Joy, Khan Raqib Mahmud, Bijan Paul, Performance Analysis of Deep Learning YOLO models for South Asian Regional Vehicle Recognition, (IJACSA) International Journal of Advanced Computer Science and Applications, ISSN:2156-5570 (Online), Vol.13, Issue.9, PP.864-873, 2022, DOI: 10.14569/IJACSA.2022.01309100.

[5] Aashish Joshi, Arni Tharakaram Hariram, K M Vishall Somaiya, Mubashir Hussain, Smart Car Parking System, International Journal of Engineering Research & Technology, ISSN:2278-0181, Vol.9, Issue.9, PP.4477-4481, Sep 2020, http://dx.doi.org/10.17577/IJERTV9IS090305.