Car Parking Spaces Detection Using Satellite Images

Maitri Mistry
Dept.of CSEE
UMBC
Baltimore, Maryland
mmistry3@umbc.edu

Manogna Lakkadasu
Dept.of CSEE
UMBC
Baltimore, Maryland
wv25053@umbc.edu

Pravalika Papasani
Dept.of CSEE
UMBC
Baltimore, Maryland
io11937@umbc.edu

***Abstract*—Nowadays, many urban environments, including our campus at UMBC, frequently face parking issues due to limited availability, leading to time-consuming searches for spots, traffic congestion, and increased carbon emissions. This has highlighted the need for efficient parking space management to save time and reduce the frustration associated with finding open parking spaces. Our project aims to develop a service that detects available parking spaces by leveraging satellite imagery sourced from Google Earth Engine and LCMS, employing advanced YOLO-based deep learning models. Our proposed solution features a user-friendly interface that identifies and displays available spots, significantly enhancing the user experience. This initiative not only aids in managing parking efficiently but also contributes to academic and technological advancements in the application of service-oriented computing in real-world scenarios.**

***Index Terms*—Parking Management, Satellite Imagery, Deep Learning**

I. Introduction

Effective parking management continues to be a significant challenge in urban areas, greatly impacting daily commutes and traffic flow. At the University of Maryland, Baltimore County (UMBC), and other large educational institutions, this issue intensifies during peak hours as the demand for parking spaces sharply increases. The difficulty in locating available parking spaces can lead to students arriving late to classes and professors to meetings, disrupting the daily schedule and productivity of the university community.

To address this issue, our project utilizes high-resolution satellite imagery—capable of covering vast areas in a single pass—combined with state-of-the-art YOLO-based deep learning models to develop a robust car parking detection system. Satellite imagery is well-suited for large-scale applications like urban planning and resource management, providing an essential comprehensive view for managing parking at large campuses or in urban areas. This technology has been adapted to specifically tackle the unique challenges associated with identifying parking spaces from satellite images, which vary significantly in terms of scale, perspective, and environmental conditions.

In addition to technical advancements, our project also focuses on developing a user-friendly interface that delivers parking availability information directly to UMBC students and staff, significantly reducing the time they spend searching for parking.

II. Related Work

The challenge of detecting parking spaces has become increasingly relevant due to its potential to mitigate urban congestion and enhance traffic management. Recent advances in satellite imagery and deep learning have significantly transformed this field by introducing new methodologies. For instance, [1] explored the use of stereo satellite images for detecting parked cars through 3D localization methods, significantly enhancing vehicle detection accuracy in large parking areas. Furthermore, the use of advanced machine learning models, particularly those based on the YOLO architecture, has revolutionized object detection tasks. [2] effectively adapted the YOLOv3 model to identify vehicles in parking lots from satellite images, utilizing transfer learning with pre-trained models to reduce the necessity for large labeled datasets while enhancing detection efficiency. This adaptability is crucial for applying these models across different observation scales and perspectives, similar to those encountered in satellite imagery. Enhancing detection mechanisms further, [3] introduced a context-enriched satellite dataset that incorporates nearby contextual data such as roads and landmarks. This integration improves the robustness of parking lot detection systems and also ensures that the systems are sensitive to the unique environment and structure of each site. The importance of real-time data in parking management has also been underscored by innovations such as the drone-based vehicle counting system developed by [4]. Building on these technological advancements, our project aims to develop a parking management system for the UMBC campus to enhance parking detection and efficiency, thereby improving urban planning and campus traffic flow.

III. Customers

A. UMBC Students

UMBC students experience a hard time parking, especially during peak hours. It saves time spent searching for parking as it shows real-time open-space availability in the parking areas. As a result, this may enable the student to attend

classes, examinations, or campus activities at the required time. Minimizing unnecessary drives by the facility supports environmental sustainability, through the burning of less fuel and emissions. Students will retrieve this through a mobile or web application, hence guaranteeing an efficient and easy way of parking.

B. Faculty/Staff

Parking issues equally affect UMBC faculty and staff during peak hours academically. It enables them to save time since, through this system, one knows exactly where free parking areas are ahead of time. They always have special areas where staff and faculty park, which this system points out where spaces are open. By streamlining parking, faculty and staff can focus on their academic and professional responsibilities undisturbed by delays associated with searching for parking spaces.

C. Visitors

Clear, real-time parking availability translates to visitors-parents, alumni, and event attendees-reducing the need to decipher a bunch of signs or even staff instructions. Instead, visitors are directly guided to available spaces. This system is particularly valuable during large events, such as open houses or graduations, where visitor traffic is at its peak. It enhances visitor satisfaction and boosts UMBC's public image as a well-organized, visitor-friendly campus by evening out vehicle distribution across lots.

D. Campus Security and Parking Services

The system is used by the parking services and school security to monitor and regulate the usage of parking lots. Real-time tracking enables the security staff to handle parking violations more effectively and divert traffic to less congested areas. The system provides insights into parking trends that help in making data-driven decisions on future infrastructure requirements and optimization of parking operations. The system helps in managing the flow of traffic and crowd during major events, thus making all users on campus have a seamless and secure experience.

IV. Methods

A. Architecture

The architecture diagram shown in Fig 1 illustrates the end-to-end workflow of a car parking detection system using machine learning. It begins with data collection and preprocessing, where satellite images are cleaned, resized, and augmented. The prepared data are then used for training and testing the model, ensuring its accuracy. Once validated, the model is deployed to the backend via a Flask API, which interacts with real-time data sources and a database for storing results. The frontend (built using React.js) fetches predictions from the backend and displays real-time parking availability to users through a user-friendly interface. This modular design ensures a seamless integration of machine learning, real-time processing, and user interaction.

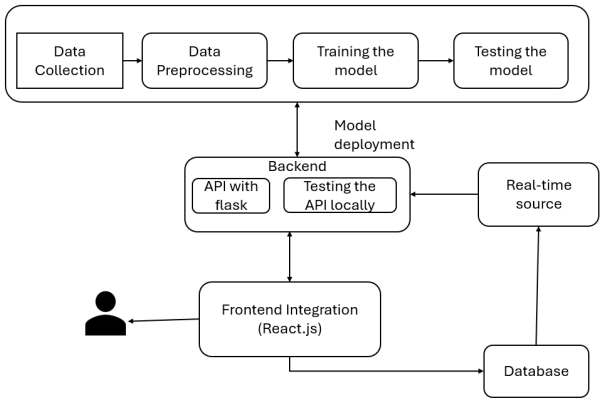

Fig. 1: Architecture Diagram

B. Data Collection

Our dataset consists of high-resolution satellite images of parking lots at the University of Maryland, Baltimore County (UMBC) campus, sourced from Google Earth Engine and the Land Cover Mapping System (LCMS). These images are collected over the past years, capturing various parking scenarios at different times of day and under diverse weather conditions. Initially, the dataset includes 150 raw images. Refer to Figure 2 for a visual representation of the dataset. Each image has been manually annotated using Roboflow to label parking spaces as 'Car' or 'Empty', ensuring precise marking and labeling of parking spaces, as illustrated in Figure 3. After extensive preprocessing and augmentation, the dataset has expanded to a total of 901 images, with 150 original and 751 augmented images. This dataset provides a robust foundation for training and validating our advanced YOLO-based deep learning models.

C. Data Preprocessing

Data preprocessing is a vital step in preparing satellite imagery for deep learning models, ensuring that raw data is transformed into a clean, diverse, and model-ready format. For this project, preprocessing was carefully designed to address variations in satellite image quality, scale, and orientation, creating a robust dataset for detecting parking spaces. The following steps outline the comprehensive preprocessing approach used:

1) Resizing and Tiling: Satellite images often vary in resolution and aspect ratio, making standardization essential for consistent model training. All images were resized to uniform dimensions to ensure compatibility with the YOLO-based models. Larger images were divided into smaller, localized tiles, enabling the models to focus on specific regions within the imagery. This tiling approach not only improved computational efficiency but also enhanced the precision of parking space detection by enabling the analysis of detailed image segments.

2) Filtering: Maintaining high-quality data was critical to ensuring the reliability of the training process. Images with

(a)

(b)

Fig. 2: Satellite images of UMBC parking lots

null or missing values were identified and removed, as such data could introduce noise and reduce model performance. By eliminating incomplete or corrupted images, the dataset integrity was preserved, allowing the models to learn effectively from meaningful, high-quality examples.

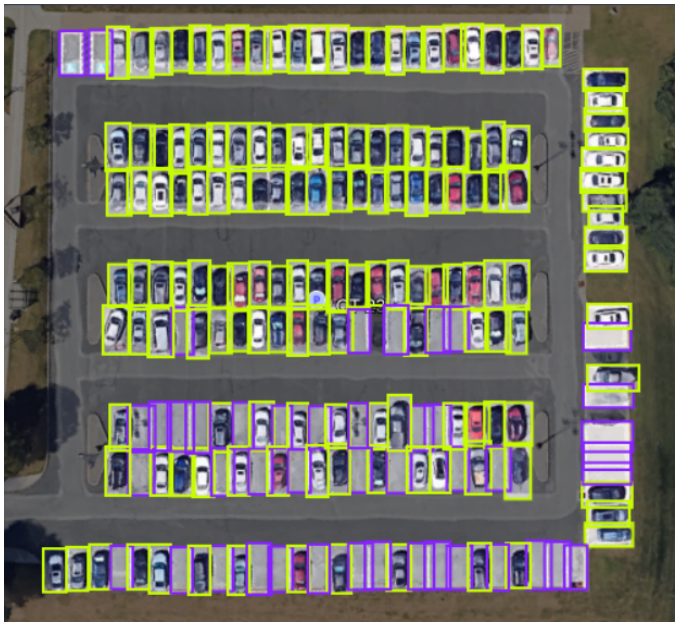

Fig. 3: Annotated image

3) Data Augmentation: To further enrich the dataset, advanced data augmentation techniques were applied, significantly enhancing its size and variability. Augmentation is essential in deep learning to simulate real-world conditions, allowing models to generalize better and adapt to various scenarios. By creating modified versions of existing images, the augmented dataset captured a wide range of environmental and structural variations in parking spaces.

The first category of augmentation focused on spatial adjustments, which addressed the orientation and positioning of parking spaces. Horizontal and vertical flips were implemented to replicate different viewing perspectives, while rotations ranging from ±15° and full 90° rotations ensured robustness to varied camera angles. These transformations helped the models recognize parking spaces regardless of their orientation in the satellite imagery.

The second category of augmentation dealt with quality adjustments, which simulated real-world imaging challenges such as lighting and sensor variations. Brightness and exposure levels were altered by up to ±15% to mimic differences in lighting conditions caused by time of day or weather. Gaussian blur, applied up to 0.7px, introduced slight focus inconsistencies, while random noise affecting 0.1% of pixels simulated imperfections in sensor data. These modifications made the dataset more realistic and prepared the models to handle environmental noise effectively.

Overall, these augmentation techniques addressed the diverse challenges of satellite imagery, ensuring that the YOLO-based models could detect parking spaces with high accuracy under a variety of conditions. The process expanded the dataset from 150 original images to 901, providing the models with a robust foundation for training and significantly reducing the risk of overfitting.

D. Models

1) YOLOv11: YOLOv11 is a powerful improvement over previous YOLO models, designed to handle complex object detection tasks with greater precision and efficiency. Its updated backbone and neck architecture make it highly effective at recognizing small objects like cars in satellite images, even in challenging environments. For instance, when dealing with parking lots, the model can navigate issues like shadows, overlaps, and varied perspectives, ensuring accurate detection. One of its key strengths is the incorporation of attention mechanisms, which help the model focus on critical areas in an image, minimizing errors and improving overall accuracy. Despite its high performance, YOLOv11 remains efficient, allowing it to process large-scale satellite imagery quickly. These features make YOLOv11 a reliable choice for our project, where precision in detecting parking spaces is critical [6], [7].

2) YOLO-NAS: YOLO-NAS takes a different approach by using Neural Architecture Search (NAS) to automatically fine-tune its design for specific tasks like parking detection. Instead of relying on manually crafted architectures, YOLO-NAS evaluates multiple configurations to find the most effective balance between accuracy and speed. This makes it particularly useful for real-time applications, such as providing up-to-the-minute parking availability updates. Its lightweight design allows it to process data quickly without overloading computational resources, making it suitable for use on devices with limited power, like mobile phones or edge devices. While it may not be as precise as YOLOv11, YOLO-NAS excels in scenarios where speed and efficiency are top priorities, offering a complementary option for our system [13].

In this project, YOLOv11 and YOLO-NAS were chosen to complement each other. YOLOv11 handles tasks where high accuracy is essential, such as identifying every parking space in densely packed or complex parking lots. Meanwhile, YOLO-NAS steps in for real-time operations, where faster response times are critical. Together, these models form the backbone of a system designed to tackle the challenges of parking space detection, ensuring both reliability and speed. By leveraging their unique strengths, we created a solution that balances precision and practicality, making parking management more efficient and user-friendly.

V. Evaluation Metrics

Standard object detection metrics used for the performance evaluation of the Parking Space Detection System are the Mean Average Precision (mAP), Precision, and Recall. These metrics are chosen because they can clearly present the accuracy and efficiency of object detection models like YOLOv11 and YOLO-NAS.

A. Mean Average Precision (mAP)

It is generally accepted as a metric of object detection, which summarizes the precision-recall curve for various Intersection over Union thresholds. Here, mAP@50 (IoU threshold of 0.5) and mAP@50-95 were evaluated, which are an average of mAP at IoU thresholds from 0.5 to 0.95 with a step size of 0.05. These values represent insight into the model's capability for highly precise object detection with good localization.

B. Accuracy

Precision estimates the ratio of parking spaces correctly identified, or true positives, out of all the spaces detected, including false positives. High precision reflects the model's reliability in keeping false alarms as low as possible-a very important factor for users relying on accurate detection to make real-time parking decisions.

C. Recall

Recall measures the ratio of the number of correctly detected parking spaces to the total number of parking spaces actually present. High recall means that most of the parking spaces were correctly identified by the model and minimizes missed detections.

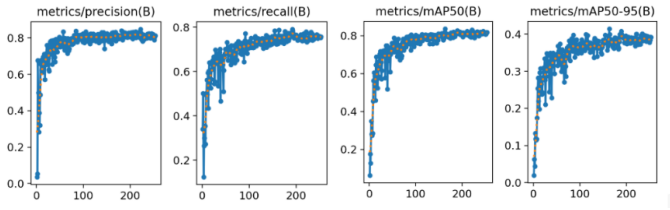

Fig. 4: Metrics

Precision, Recall, and mAP at various thresholds were used for performance evaluation of the YOLOv11 model. Precision refers to the fraction of the correctly detected parking spaces with respect to all detections, indicating how well the model avoids false positives. Recall gives the proportion of true parking spaces that were correctly detected, hence ensuring that the identifications are comprehensive with a minimum number of missed detections. The mAP50 metric, evaluating the detection accuracy with an IoU threshold as lenient as 50%, got stabilized above 0.8, reflecting a good performance on robust detection. mAP50-95 is a metric that evaluates performance even more stringently over IoU thresholds from 50% to 95%, and values are around 0.4, demonstrating adaptability under different detection criteria. Collectively, these metrics underline the strong aptitude of the model in producing highly accurate and reliable parking space detection, suitable for real-world deployment in dynamic scenarios.

VI. Results

For the performance evaluation of the parking space detection system, advanced deep learning models such as YOLOv11 and YOLO-NAS were utilized. Both are chosen because of their superior capabilities in object detection with accuracy and efficiency.

YOLO (You Only Look Once) represents one of the most classical and highly performing one-stage frameworks

in object detection [6], [7]. Herein, YOLOv11 improves architectural novelty and allows for enhanced feature extraction capabilities through advanced convolution blocks together with attention mechanisms; as a result, it gives even better detection performance, including in complicated scenarios [6]. Therein, it keeps up real-time performances with detection accuracy to handle real-world parking lot scenarios.
YOLO-NAS: YOLO-NAS represents an optimized version of YOLO with Neural Architecture Search (NAS). It automates the discovery of efficient neural network architectures that optimize precision while reducing computational costs. Among others, YOLO-NAS focuses on the trade-off between speed and accuracy, especially for edge devices and real-time inference tasks [13].
Standard metrics were employed in the evaluation of both models using the Mean Average Precision (mAP), Precision, and Recall. The table summarizes comparative results achieved during the test:

Model/Metrics	mAP (%)	Precision (%)	Recall (%)
YOLOv11	83.6	82.5	78.2
YOLO-NAS	76.2	81.4	72.9

TABLE I: Model Performance Comparison for Parking Space Detection

From the results, YOLOv11 performed best in all aspects, reaching an mAP of 83.6%, Precision of 82.5%, and Recall of 78.2%, probably due to its heightened ability to detect parking spaces much more accurately and confidently. At the same time, YOLO-NAS achieved an mAP of 76.2%, Precision of 81.4%, and Recall of 72.9%, thus showing a little compromise in accuracy but still competitive in terms of precision.

The results show that YOLOv11 is more effective for real-time parking space detection since its improved accuracy is helpful in reducing both false positives and false negatives. YOLO-NAS, though a bit less accurate, is also a good alternative due to its lightweight architecture and computational efficiency.

VII. User Interface

The UI of the Parking Space Detection System shall provide an intuitive, user-friendly experience that allows users to interact with the system seamlessly, whether they be students, faculty, staff, or visitors. Modern web development technologies are used throughout the system to ensure a very responsive, real-time updated system accessible on desktop and mobile devices [7].

This system is achieved with the UI that is user friendly and easily accessible using React.Js as the front end framework [6]. The component based architecture of React will help in creating reusable components that enhance scalability and maintainability of the system. It will provide a clean and modern layout for clear labels, buttons, dropdowns and file input fields so that users can easily access the model to upload images of parking lots and detect the empty spaces.

It acts as an intermediary between the front end and the model API, powered by Flask, a lightweight Python web framework [4]. Flask provides RESTful API endpoints for easy data exchange: handling uploads of images, processing the requests, and sending results of the detections back to the UI in real time.

The following key elements have been integrated into the user interface:

Interactive Image Upload: Users upload an image of parking lots. A drag-and-drop file input that is intuitive or a ”Choose File” button allows one to select a file whose file name is shown instantly on being selected.
Detect Button: Once an image is uploaded, a user can press the button Detect Parking Spaces. In response, this sends a request to the Flask back-end, which processes the image and sends it to the machine learning model for processing.
Results: The output space shows the original uploaded image with overlaid bounding boxes for every parking space it detects, wherein each has been labeled against the two categories-Empty and Car-the model is sure about. Also, further results include additional information: detected parking lot spaces total in number, available spot count.
Responsiveness of design: The interface for this system is fully responsive for different screen sizes that start from mobile phones to a tablet and desktops; users will be able to receive information about parking availability because these modules are compatible with them.
Loading Indicators and Error Handling: While the detection process is in progress, it displays loading indicators or status messages, which keeps users informed. In the case of an error-a failure of the API, network issues, or wrong image upload-it would provide clear and actionable error messages to users.
Theme and Aesthetics: The UI design follows a minimalist approach with the primary concern of clarity and simplicity. The color scheme uses shades of blue, white, and yellow to enhance contrast and highlight key buttons such as ”Choose File” and ”Detect Parking Spaces.” Font styles and button sizes were selected to make the interface accessible to everyone.
Security and Data Privacy: In securing users' data, image uploads are securely processed on the server-side; images are not persisted anywhere. All sensitive data exchanged between front and back end are via HTTPS.

Technologies Used

- Front-End: React.js, HTML5, CSS3, JavaScript

- Back-End: Flask (Python), REST API for image processing
- Image Upload & Processing: FileReader API (client-side) and FormData API for server-side processing
- Visualization: HTML5 Canvas for drawing bounding boxes on the image
- Styling: Custom CSS for responsiveness and cross-platform compatibility

VIII. Conclusion

The UMBC Parking Space Detection System solves a lot of campus parking problems through the usage of image detection and machine learning technologies. With this system, parking is much easier because students, faculties, staffs, and visitors get updates on space availability in real time. Our model will reduce the search time and congestion while finding a parking spaces while helping the campus security and parking services to improve their tracking and manage the traffic on-campus. While the currently developed prototype has been representative in demonstrating the potential of such a system, future renditions are planned to cover real-time data processing with increased accuracy and scalability of the system. This approach demonstrates how intelligent automation enhances smart campus initiatives, guarantees operational efficiency, and significantly improves the experience of stakeholders on campus.

References

[1] Zambanini, S., Loghin, A. M., Pfeifer, N., Soley, E. M., & Sablatnig, R. (2020). Detection of parking cars in stereo satellite images. Remote Sensing, 12(13), 2170.

[2] Kumar, S., Thomas, E., & Horo, A. (2019, September). Identifying Parking Lots from Satellite Images using Transfer Learning. In 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT) (Vol. 1, pp. 1-8). IEEE.

[3] Yin, Y., Hu, W., Tran, A., Kruppa, H., Zimmermann, R., & Ng, S. K. (2022). A context-enriched satellite imagery dataset and an approach for parking lot detection. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 1371-1380).

[4] Hsieh, M. R., Lin, Y. L., & Hsu, W. H. (2017). Drone-based object counting by spatially regularized regional proposal network. In Proceedings of the IEEE international conference on computer vision (pp. 4145-4153).

[5] Google Earth Pro, "Imagery Data for Parking Lot Analysis," Google Earth, Accessed: Sep. 2023. [Online]. Available: https://www.google.com/earth/

[6] Khanam, R., & Hussain, M. (2024). YOLOv11: An overview of the key architectural enhancements. arXiv preprint arXiv:2410.17725.

[7] Zhang, Y., Guo, Z., Wu, J., Tian, Y., Tang, H., & Guo, X. (2022). Real-time vehicle detection based on improved yolo v5. Sustainability, 14(19), 12274.

[8] Vadivel, M., Murugan, S., Ramamoorthy, S., Archana, V., & Sankarasubbu, M. (2019). Detecting parking spaces in a parcel using satellite images. arXiv preprint arXiv:1909.05624.

[9] Gopinath, B., Gokul, K. S., Pumenitha, S. T., & Vasanth, S. H. (2023, June). Deep Learning based Automated Parking Lot Space Detection using Aerial Imagery. In 2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA) (pp. 1-4). IEEE.

[10] Xu, Z., Tang, X., Ma, C., & Zhang, R. (2024). Research on parking Space Detection and Prediction Model based on CNN-LSTM. IEEE Access.

[11] Grbić, R., & Koch, B. (2023). Automatic vision-based parking slot detection and occupancy classification. Expert systems with applications, 225, 120147.

[12] Hattale, P., Jangam, V., Khilare, S., Ratnaparkhi, Y., & Kasture, P. Parking Space Detection Using Image Processing. Int. J. Sci. Res.

[13] Terven, J., Córdova-Esparza, D. M., & Romero-González, J. A. (2023). A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. Machine Learning and Knowledge Extraction, 5(4), 1680-1716.