Drive Safe360 - AI Based Tint Detection and Parking Allocation

Ashwani Sangwan[1], and Devansh Chaudhary[1]

Abstract

With the increasing demand for secure and compliant parking solutions in public and institutional areas, there is a need for intelligent vehicle screening systems. This paper presents an AI-based model that first detects incoming vehicles using YOLOv3, identifies the presence of tinted windows using a Convolutional Neural Network (CNN), and then conditionally allocates parking based on the presence or absence of tinting. In scenarios where tinted windows are not detected, the system calculates available parking spots using camera feeds and assigns them dynamically. Due to the lack of public datasets for tint detection, a custom dataset was created via web scraping. This project offers a robust solution to compliance-based parking access and contributes to real-time smart surveillance systems.

Keywords: Image Analysis, Tint Detection, YOLOv3, Convolutional Neural Network, Real-Time Processing, Vehicle Compliance.

1. Introduction

1.1 Project Background

Urban environments face mounting challenges due to the ever-increasing number of vehicles. Parking management has become a significant issue in large campuses, public spaces, and office complexes. Traditional parking systems often rely on manual monitoring, ticketing, or at best, license plate recognition. These systems, while functional, lack the capacity to enforce legal vehicle compliance such as window visibility laws. With rising security concerns and regulatory mandates, there is a growing need to not only monitor but also enforce vehicular compliance at entry points. In recent years, Artificial Intelligence (AI) and Computer Vision have made significant strides in automating these processes. Object detection models like YOLO (You Only Look Once) and classification models powered by Convolutional Neural Networks (CNNs) have enabled real-time vehicle monitoring. However, their integration into enforcement mechanisms, especially those targeting regional legal compliance like tint detection, remains largely unexplored.

1.2 Project Description and Legal Context

In India, vehicle window tinting is strictly regulated. As per the Supreme Court ruling dated April 27, 2012, and Rule 100 of the Central Motor Vehicles Rules (CMVR), 1989, the use of black films or tinted glass with less than 70% Visual Light Transmission (VLT) for the front and rear windshields, and less than 50% VLT for side windows, is prohibited. These laws were enacted to enhance visibility into vehicles for law enforcement and ensure passenger safety. However, enforcement is mostly reactive and manual, relying on on-road police inspection. This project proposes a proactive solution by integrating an AI-powered system capable of detecting window tint in real time. It acts as a compliance checkpoint by scanning incoming vehicles, detecting the presence of tint, and determining access permissions based on the results. This solution is particularly useful in secure parking environments such as corporate campuses, universities, and government buildings where entry control is necessary.

1.3 Gap in Existing Solutions and Challenges Addressed

While vehicle detection has been extensively researched and implemented, particularly through the YOLO family of models, there is a visible gap in literature and practice when it comes to vehicle attribute detection—especially for legally regulated features such as window tint. Most smart parking systems only focus on parking slot availability, license plate detection, or RFID-based access. Very few, if any, consider regulatory compliance as a prerequisite for granting parking access. Another significant challenge is the lack of publicly available datasets for car window tint detection. This limitation prevents researchers and developers from training robust models that can perform this task in real-world scenarios. Our work addresses this challenge by curating a custom dataset through extensive web scraping and manual labeling. It simulates real-world conditions like variable lighting, reflections, and different car models, enhancing model robustness.

1.4 Research Contribution and Novelty

This project presents an end-to-end AI system that integrates vehicle detection, window tint classification, and real-time parking slot management. The primary contributions are: (i) development of a novel dataset for car window tint detection, (ii) a CNN-based model trained to classify tinted vs. non-tinted vehicles with high accuracy, (iii) integration of YOLOv3 for real-time vehicle detection, and (iv) dynamic parking space assignment based on compliance.

Unlike existing solutions that focus solely on vehicle detection or static parking allocation, our system ensures regulatory compliance by identifying tinted vehicles and restricting their access to secure or surveillance-critical parking zones. The proposed model is capable of processing live camera feeds, allowing real-time decision-making and automation without human intervention. A lightweight architecture and optimized inference pipeline make it deployable on edge devices or embedded systems, enabling cost-effective implementation.

Furthermore, the dataset curated for this study fills a crucial research gap, as no publicly available datasets were found to address the tint detection problem. Our dataset emphasizes a wide range of lighting conditions, camera angles, and vehicle types to ensure robust performance in diverse environments. Additionally, a Power BI dashboard was developed to visualize parking slot availability, vehicle entry logs, and compliance statistics, enhancing system transparency and management efficiency.

2. Literature Review

Vehicle detection, classification, and automated compliance systems have gained momentum in the era of smart cities. Various deep learning frameworks have been proposed to optimize traffic flow, detect vehicular features, and automate parking operations. This section reviews key contributions that align with the scope of our proposed solution.

Asha et al. (1) presented a YOLO-based real-time vehicle counting model integrated with a correlation filter for traffic management systems. Their study emphasized the need for rapid object detection, which is a foundational element in our car detection mechanism. However, their work was limited to vehicle counting and lacked integration with access control or parking assignment logic.

Gomaa et al. (3) proposed a robust vehicle detection and counting framework combining convolutional neural networks (CNN) with optical flow techniques. Their approach focused on dynamic scenes, overcoming challenges like occlusion and varying traffic density. While their method achieved high accuracy, it was primarily designed for highway traffic analysis and lacked extensibility for use cases such as tint compliance or smart parking allocation.

Bochkovskiy et al. (10) introduced YOLOv4, which brought significant improvements in speed and accuracy compared to earlier versions. YOLOv3, used in our project, was selected due to its optimal balance between computational efficiency and detection accuracy, especially suitable for real-time deployment in campus or office environments.

Kaliyaperumal (4) proposed an IoT-enabled vision system specifically for detecting tinted windows in vehicles. The paper analyzed tint detection using image processing techniques and evaluated compliance against regional regulations. While the study tackled a closely related problem, it did not incorporate real-time vehicle tracking or parking space allocation, which are novel additions in our approach.

Jmour et al. (8) analyzed the role of CNNs in image classification and object feature extraction. Their work laid the groundwork for various applications, including our CNN-based tint classifier. The power of convolutional layers in distinguishing visual features is particularly effective in differentiating tinted versus non-tinted windows, which can often have only subtle contrast differences.

Wu et al. (11) developed the Cascade R-CNN model for high-quality object detection through multi-stage refinement. While offering superior detection accuracy, their solution is resource-intensive. Our decision to use YOLOv3 instead ensures that we maintain real-time responsiveness while still achieving satisfactory detection accuracy.

These studies highlight major advancements in object detection, vehicular feature analysis, and smart surveillance. However, there is a clear gap in integrating these technologies into a unified pipeline that enforces tint compliance, performs car detection, assigns parking spaces dynamically, and monitors overall occupancy. Our proposed work bridges this gap by combining detection, classification, token generation, and visualization within a practical and scalable framework powered by YOLOv3, CNNs, Flask, Power BI, and PySpark.

3. Methodology

The proposed system is designed to automate the process of vehicle detection, tint classification, and dynamic parking space allocation in a controlled environment such as an office, campus, or public parking facility. The system integrates two major components: (i) an AI-based visual analysis pipeline, and (ii) a parking slot management logic. Figure 1 illustrates the overall system architecture.

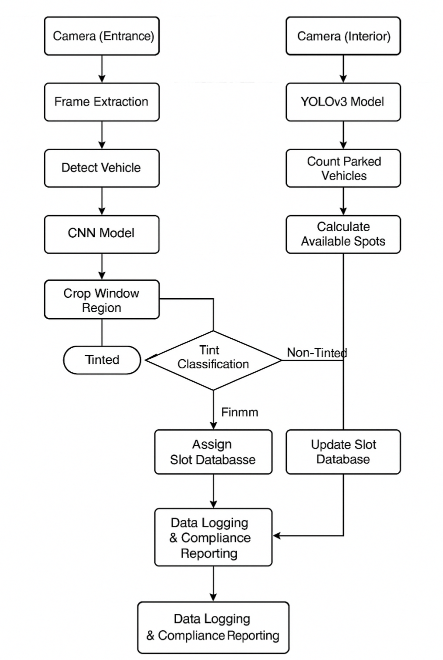

Figure 1: System Architecture

3.1 Dataset

3.1.1 Challenges in Dataset Availability

One of the primary challenges encountered during the development of this project was the unavailability of an open-source or public dataset specifically designed for detecting car window tint levels. While several large-scale datasets exist for general object detection tasks—such as the COCO dataset and Open Images Dataset—none of them include annotations related to vehicle window transparency or tinting. This posed a significant roadblock for training a machine learning model capable of distinguishing between tinted and non-tinted vehicle windows.

Detecting window tint is a nuanced task due to variations in lighting conditions, window reflections, camera angles, and car models. Moreover, defining the degree of tint from images alone is inherently subjective without light transmission metrics (like VLT%), which are not available in standard image datasets. Therefore, traditional approaches relying on labeled datasets or numerical compliance thresholds were not applicable. This created the necessity to construct a purpose-built dataset tailored for our specific use case.

3.1.2 Custom Dataset Creation Process

To address this gap, we undertook the task of building a custom dataset from scratch. High-quality images of cars were scraped from various online sources including automotive websites, surveillance footage samples, car review portals, and image-sharing platforms. The scraping process focused on capturing diversity in:

- Angle of view (front, side, rear)
- Window state (clear vs tinted)
- Lighting conditions (daylight, low-light, indoor)
- Vehicle types (sedan, hatchback, SUV)

Over 1000 images were collected initially, which were then manually filtered and annotated. Each image was labeled into one of two classes: "Tinted" or "Non-Tinted" based on visible darkness and opacity of the side windows, verified by human annotators. We intentionally excluded overly ambiguous samples to improve model learning accuracy and reduce misclassification during training.

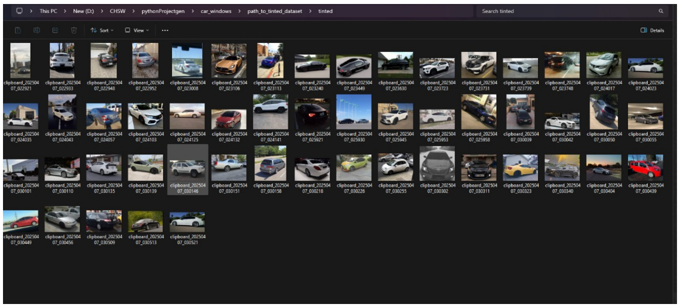

Figure 2: Tinted Dataset Samples

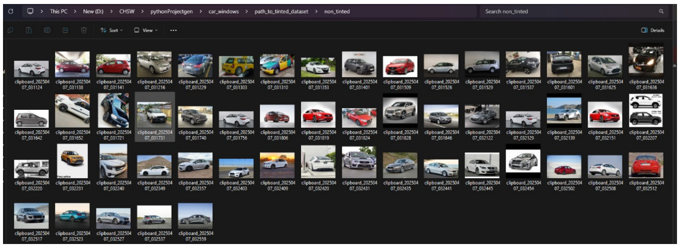

Figure 3: Non-Tinted Dataset Samples

3.1.3 Dataset Structure and Use

The final curated dataset consisted of around 950 usable images—balanced across the two classes—to be used for training and validating the CNN-based tint classification model.

Annotations were stored in a CSV format, mapping image filenames to their respective class labels. The dataset has been structured for easy reuse and extension, with potential for future enhancements such as incorporating bounding box data or degree-of-tint estimation using regression models.

3.2 Input Acquisition and Preprocessing

The system receives live video input through a static surveillance camera installed at the entrance of the parking facility. The camera feed is broken down into frames at a predefined rate (e.g., 5 frames per second) to ensure efficient processing while maintaining near real-time performance. Each frame undergoes basic preprocessing such as resizing, noise filtering, and normalization before being passed to the object detection model.

3.3 Vehicle Detection using YOLOv3

Each preprocessed frame is passed through the YOLOv3 (You Only Look Once, Version 3) object detection model. YOLOv3 is selected due to its balance of speed and accuracy, which makes it suitable for real-time applications. The model has been pre-trained on the COCO dataset and fine-tuned on additional custom samples to enhance car detection performance in varied lighting and environmental conditions.

- **Output:** YOLOv3 returns bounding boxes, class labels (e.g., ”car”), and confidence scores for all detected vehicles in the frame.
- **Post-processing:** Non-Maximum Suppression (NMS) is applied to eliminate overlapping detections. The bounding box coordinates of each detected car are extracted for the next stage.

3.4 Tint Detection using CNN

From the YOLO-detected car bounding boxes, cropped images of vehicle side windows are extracted and passed to a custom-trained Convolutional Neural Network (CNN). This CNN has been designed and trained specifically for classifying whether the side windows of a car are tinted or non-tinted.

- **Input Format:** Cropped side-window image (preprocessed and resized, typically 224x224).
- **CNN Architecture:** A relatively lightweight CNN was chosen to ensure low latency. It consists of:
  - Convolutional layers with ReLU activation
  - MaxPooling layers for spatial reduction
  - Dropout for regularization
  - Fully connected layers ending in a Softmax output for binary classification (Tinted / Non-Tinted)
- **Training Data**: The CNN was trained on a custom dataset built by scraping and annotating over 950 images, augmented using techniques like flipping, brightness shift, and cropping to improve generalization.
- **Output:** A classification label (Tinted / Non-Tinted) with a confidence score.

3.5 Decision Logic: Access Control & Parking Assignment

Based on the tint classification result, the system implements the following logic:

- **If the vehicle is tinted:** The vehicle is granted entry **without** assigning a parking slot. This design simulates scenarios where tinted vehicles are allowed to freely enter but not occupy monitored spaces, or perhaps have reserved VIP parking.
- **If the vehicle is not tinted:** The system checks for **available parking slots** and **assigns a slot** to the vehicle.

This ensures that only compliant vehicles are integrated into the managed parking system.

3.6 Real-Time Parking Slot Estimation

To dynamically manage parking space allocation, the system monitors a camera feed from inside the parking area. The following steps are applied:

- **Vehicle Count:** YOLOv3 (or another instance of the same model) is used to count the number of parked vehicles in the current frame.
- **Slot Estimation:**
  - Let T = Total parking slots available
  - Let C = Count of currently parked vehicles
  - Then, **Available slots = T - C**
- **Database Update:** Once a non-tinted vehicle is assigned a slot, the system updates the internal slot database (which could be a simple JSON or SQL table) to mark the assigned slot as occupied. Upon exit (detected by another exit camera), the slot is marked as free.

3.7 Logging and System Feedback

Each decision, including vehicle detection, tint classification result, parking slot status, and timestamp, is logged into a central server or local database. This serves two purposes:

- **Compliance reporting:** Allows for audit and analytics of compliance behavior (e.g., what percentage of vehicles are tinted).
- **System monitoring:** Facilitates debugging, historical data tracking, and visualization on an admin dashboard.

4. Results and Discussions

The proposed system integrates real-time vehicle monitoring, AI-based compliance analysis, intelligent parking management, and data visualization into a seamless and user-friendly interface. The system was evaluated in simulated and semi-real environments representing campus and institutional parking zones. This section presents the functional and analytical outcomes of the deployment.

CarID	Colour	Parking_Zone	Stay Duration(Hours)
MU5076	White	A1	4.12
IF3713	White	A5	4.01
YZ2385	Green	A3	4.75
SQ4055	Green	A1	4.83
MC4459	Green	A1	2.46
MO4573	Green	A1	2.71
UO1423	Green	A1	2.44
KB1292	Green	A1	3.46
AG1916	Green	A1	4.62

Figure 4: Car Entry Dashboard

4.1 Dashboard-Based Monitoring System

A web-based dashboard was developed using **Flask** for backend handling and **HTML/CSS** for the frontend interface. The dashboard acts as the control panel for system administrators and has four distinct observational modules:

- **Live Vehicle Entry Monitor**: Displays detected vehicles with labels (tinted/non-tinted), timestamps, and live camera feed snapshots.
- **Token-Based Ticket Allocation**: Assigns a unique random token to each non-tinted vehicle upon entry, logged in a backend database.
- **Zone-Wise Parking Estimation**: Uses YOLOv3 to count parked vehicles in each zone and subtracts from total slots to determine live availability.
- **Data Insights & Analytics**: Processes vehicle data using PySpark and visualizes trends via Power BI dashboards.

4.2 System Workflow and Architecture

The architecture of the system integrates car detection, tint classification, parking logic, and real-time logging. Detection is performed via YOLOv3 for vehicles and a CNN model for tint verification. Data is passed to the dashboard and backend services.

4.3 Random Token Generation Mechanism

For non-tinted vehicles, a random parking token is issued to avoid redundancy and manual entry. Each token is a 6-character alphanumeric code generated using:

```
id = ''.join(random.choices(string.ascii_uppercase + string.digits, k=6))
```

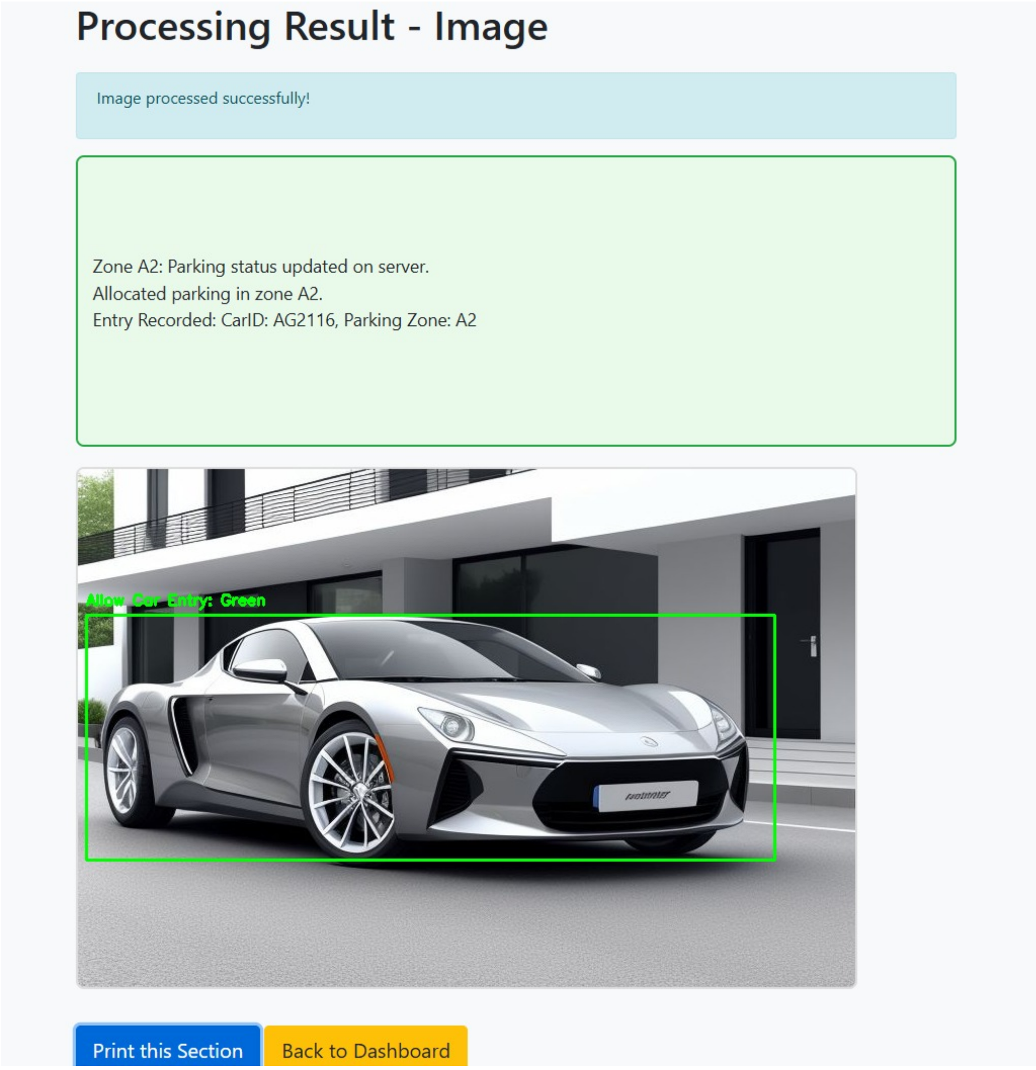

Figure 5: Token Based Allocation

Figure 6: Analysis of Stay Duration/Fare

4.4 Real-Time Parking Slot Estimation

Each parking zone is monitored by static cameras. YOLOv3 detects cars in real time, and the number of occupied spots is updated:

$$Vacancy_{zone} = Total_{zone} - Occupied_{zone} \tag{1}$$

This live decrementing mechanism ensures availability accuracy. Each vehicle entry triggers a real-time update of the respective zone's availability.

4.5 Data Processing and Visualization using PySpark and Power BI

Raw logs are pre-processed using PySpark to transform data into vehicle categories, entry times, tint status, and parking durations. This structured data is sent to Power BI for generating real-time analytics dashboards.

Key insights include:

- Tint vs Non-Tint entry ratios
- Peak traffic hours
- Zone-wise occupancy distribution
- Duration-based heatmaps

4.6 Performance Summary and Observations

Scalability: The architecture supports scaling across multiple zones and entry gates. Flask APIs and YOLO models can be containerized for larger deployments.

Compliance: The system ensures adherence to regional tint laws by rejecting tinted vehicles and logging violations for reports.

Usability: The dashboard provides a non-technical interface for real-time monitoring and manual override.

Data Logging: Every entry, detection, allocation, and exception is logged with a timestamp for traceability and compliance audit.

—

References

[1] C. S. Asha, V. Mandal, and M. Landowski, "Vehicle counting for traffic management system using YOLO and correlation filter," in *Proc. IEEE Int. Conf. Electron., Comput. Commun. Technol. (CONECCT)*, Bangalore, India, 2018, pp. 1–6.

[2] A. Galletta, G. Plastiras, C. Kyrkou, and T. Theocharides, "MeSmart-Pro: Advanced processing at the edge for smart urban monitoring and reconfigurable services," *J. Sensor Actuator Netw.*, vol. 9, no. 3, pp. 34–45, 2020.

[3] A. Gomaa, J. A. Arinaldi, and A. A. Gurusinga, "Robust vehicle detection and counting algorithm employing a convolution neural network and optical flow," *Sensors*, vol. 19, no. 20, p. 2407, 2019.

[4] G. Kaliyaperumal, "IoT-enabled vision system for detection of tint level," *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, no. 4, pp. 2388–2397, 2024.

[5] M. Mohana, R. Kejriwal, R. H. J., and A. Arora, "Artificial Intelligence (AI) enabled vehicle detection and counting using deep learning," in *Proc. Int. Conf. Comput. Commun. Informat. (ICCCI)*, Coimbatore, India, 2022, pp. 1–8.

[6] A. A. Permana and A. R. Lestiawan, "Vehicle object detection and classification using machine learning algorithm in Tangerang City," *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, no. 4, pp. 2388–2397, 2024.

[7] Y. Xia, O. P. Barus, and I. Delano, "Towards improving quality of video-based vehicle counting method for traffic flow estimation using virtual loop and Gaussian mixture models," *Signal Process.*, vol. 120, pp. 672–681, 2016.

[8] N. Jmour, S. Zayen, and A. Abdelkrim, "Convolutional neural networks for image classification," in *Proc. IEEE Int. Conf. Intell. Inf. Hiding Multimedia Signal Process. (IIH-MSP)*, 2018, pp. 1–5.

[9] J. Bathija, "Visual object detection and tracking using YOLO and SORT," *Int. J. Eng. Res. Technol.*, vol. 8, no. 11, pp. 705–708, 2019.

[10] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[11] J. Wu, L. Cao, K. He, and J. Sun, "Cascade R-CNN: Delving into high-quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 6154–6162.

[12] A. B. Amjoud and M. Amrouch, "Object Detection Using Deep Learning, CNNs and Vision Transformers: A Review," *IEEE Access*, vol. 11, pp. 35479–35516, 2023.

[13] T. Turay and T. Vladimirova, “Toward Performing Image Classification and Object Detection with Convolutional Neural Networks in Autonomous Driving Systems: A Survey,” *IEEE Access*, vol. 10, pp. 14076–14119, 2022.

[14] K. Sharma, T. K. Sahu, P. Bhatt, A. K. Singh, and M. Bhatia, “Real Time Video Object Detection Using Deep Learning,” in *Proc. Int. Conf. Adv. Comput., Commun. Inf. Technol. (ICAICCIT)*, Faridabad, India, 2023, pp. 601–606, doi: 10.1109/ICAICCIT60255.2023.10466207.

[15] S. Chen and Y. Zhang, “Half-UNet: A simplified U-Net architecture for medical image segmentation,” *Front. Neuroinform.*, vol. 16, Article 911679, 2022.