Volume 9, Issue 9, September– 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP1017

YOLOv8 Powered Solutions for Box Identification in Warehouses

S. Thaneesan[1*]; J. A. K. S. Jayasinghe[2] (Professor)
[1]Department of Science and Technology, Faculty of Applied Sciences, Uva Wellassa University of Sri Lanka, Badulla, Sri Lanka.
[2]Department of Electronic & Telecommunication Engineering, Faculty of Engineering, University of Moratuwa, Sri Lanka.

Correspondence Author:- S. Thaneesan[1*]

Abstract:- In modern warehouse management, the ability to effectively identify and track boxes is critical for optimizing operations and reducing costs. This research investigates the application of YOLOv8 deep learning model for real-time box identification in warehouse environments. Three different approaches were evaluated: using a pre-trained YOLOv8 model, training the model with a dataset obtained from the Internet, and training the model with a custom dataset designed for this application. For the second and third approaches, the model was trained using Google Colab, and image annotation was performed using Roboflow. Each approach is thoroughly tested to assess the accuracy and robustness of the model under various conditions. The results demonstrate the strengths and limitations of YOLOv8 in different scenarios, providing valuable insights into its practical implementation for warehouse automation. This study highlights the potential of YOLOv8 as a useful tool for improving warehouse efficiency.

Keywords:- *YOLOv8, Google Colab, Roboflow.*

I. INTRODUCTION

The rapid growth of the global supply chain has placed significant demands on warehouse operations, leading to many challenges that modern warehouses must address. With the advent of Industry 4.0, there is a greater emphasis on automation and adoption of cutting-edge technological advancements [1]. This paradigm shift has introduced various applications of artificial intelligence, and automation technology in smart logistics warehouses [2]. The concept of smart warehouses, also referred to as Warehouse 4.0 or Industry 4.0, aims to improve overall service quality, productivity and efficiency, while reducing costs and reducing errors.

Warehouses typically perform four distinct functions: receiving, storing, picking, and shipping [1]. Among these, the identification and monitoring of compartments is critical to ensure smooth operations. Traditional inventory management methods, such as manual scanning and barcode systems, are labor-intensive, time-consuming and error-prone. In contrast, the advancement of computer vision and deep learning technologies offers a promising solution to overcome these limitations.

Under normal circumstances, the human brain can easily recognize boxes in different contexts. However, the need for automated solutions that can accurately identify boxes in various warehouse systems has led to the development of deep learning-based detection algorithms. Algorithms such as Faster R-CNN, SSD, and YOLO have been explored for this purpose, each offering different advantages [2]. Among these, the YOLO (You Look Only Once) algorithm stands out for its convenience and performance in real-time object detection [3].

In this study, the focus is on evaluating the performance of YOLOv8 for box identification in warehouses. The primary objective is to evaluate how YOLOv8 performs under different approaches, including a pre-trained model, a publicly available dataset, and a custom dataset. This paper describes the entire process, from setting up the model to its final evaluation, providing insights into the practical applications and capabilities of YOLOv8 in a warehouse environment.

II. METHODS

- *Yolov8 Model Architecture*

The YOLO (You Only Look Once) models are popular for their accuracy and compact size [4]. Table 1 represents a comparative analysis of all YOLO versions across years of division and improvements.

YOLOv8 - the latest iteration of the YOLO series of deep learning algorithms for object detection and image segmentation developed by Ultralytics - further improved the object detection performance capabilities of its predecessors based on the latest advances in deep learning and computer vision [4], [5]. YOLOv8 delivers exceptional performance in both speed and accuracy. Its streamlined architecture makes it versatile and easily adaptable to various hardware platforms, making it an ideal choice for a wide range of applications. YOLO detection algorithms are well-suited for deployment on edge devices with limited computing power. By modifying network structures, we can improve both detection speed and efficiency in these constrained platforms and further reduce model parameters [6]. In this study, the

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP1017

YOLOv8n model is specifically used for the box detection task in warehouses. YOLOv8n, the smallest version of the YOLOv8 model has 3.2 M parameters [4]. Based on our past experience we believe that, it is particularly suitable for real-time box identification in warehouses, providing efficient performance due to its low computational resource requirements.

Table 1 Comparison of the YOLO Versions [3]

YOLO Version	Year	Improvements from YOLOv2 to YOLOv8
YOLOv2	2017	Included higher resolution and anchor boxes
YOLOv3	2018	Performance on smaller objects
YOLOv4	2020	Optimal speed and accuracy
YOLOv5	2020	Introduced mosaic augmentation
YOLOv6	2021	Architectural improvements
YOLOv7	2021	Infers faster and greater accuracy
YOLOv8	2023	Improved adaptive training, customizable, architecture, advanced data augmentation

One of the most significant features of YOLOv8 is its ability to easily convert a trained model into formats suitable for deployment in various software applications and hardware devices. This flexibility is particularly valuable for embedded system design, as it streamlines the integration process and enhances operational speed. Common YOLOv8 export formats include ONNX, PyTorch, TorchScript, TensorRT, CoreML, and PaddlePaddle, making it highly adaptable across different platforms and environments [4].

➢ *Image Annotation*

Image annotation plays an important role in the image processing pipeline, and Roboflow provides an efficient platform for this purpose. From data collection and pre-processing to model training, Roboflow simplifies the entire workflow. It provides powerful tools and features that simplify the dataset annotation and management process, making it an invaluable asset for researchers and developers [7]. By uploading, annotating and labeling images, users can easily create the required files and share datasets into training, testing and validation sets. In addition, Roboflow's platform generates code for seamless dataset integration, facilitating a smooth training process.

➢ *Model Training*

Model training, especially for complex models, can be challenging as it requires more time when using CPUs on standard laptops or computers. Google Colab addresses this challenge by significantly speeding up the training process through powerful GPUs and TPUs, with free access giving limited performance and the option to upgrade to Google Colab Pro for enhanced performance with additional resources. In this research, the T4 GPU at Google Colab was used to train the YOLOv8 model, which enables efficient processing and optimization. At the end of training, model files and weight files are generated, which can be used for making predictions on new data or exported to various formats mentioned in Section 0.

In this study, the YOLOv8n model was trained using Google Colab for Approach 2 and Approach 3 described below (Note: Approach 1 needs no training). All inference tasks for the three approaches were performed on a local laptop. The inference process was facilitated by setting up a Python environment in Visual Studio Code (VS Code) on the laptop. The environment was built with Python 3.12 and pip 24.1 to ensure compatibility and performance during the inference phase.

- *Approach 1: Using a Pre-Trained YOLOv8n Model.*

In this approach, the pre-trained 'yolov8n.pt' model was directly downloaded and used for box detection.

- *Approach 2: Using a dataset obtained from the Internet.*

For this approach, a publicly available dataset from Roboflow Universe [8] was used to train the YOLOv8n model. **Error! Reference source not found.** shows a sample annotated image used in this approach. This dataset contains 2,358 images; however, a significant drawback is the presence of many unwanted boxes that are not suitable for warehouse environments. The training process was conducted at Google Colab using its T4 GPU. The number of epochs was set to 100. After training, the model file was generated. This model was subsequently tested on a local laptop to evaluate its performance in box detection.

Fig 1 Sample Annotated Image for used in the Second Approach

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP1017

- *Approach 3: Using a custom dataset Designed for this Application.*

This approach involves creating a custom dataset designed exclusively for warehouse box identification under various lighting conditions. Images were collected and annotated using Roboflow, followed by partitioning of the dataset into training, test and validation sets. **Error! Reference source not found.** 2 shows a sample annotated image used in this approach. The YOLOv8n model was trained on Google Colab, similar to the second approach, with the number of epochs set to 100. The performance of the trained model was then analyzed by running inference using the generated model file on a local laptop.

Fig 2 Sample Annotated Images from the Custom Dataset: (a) Image Obtained from the Internet and Annotated using Roboflow, (b) Image taken onsite at the warehouse and Annotated using Roboflow.

III. RESULTS AND DISCUSSION

In this study, the performance of the YOLOv8 model was evaluated with three different approaches to identify boxes in a warehouse environment. Each approach produced different results.

In the first approach, which involves using a pre-trained YOLOv8 model, the performance is not satisfactory for detection of boxes in a warehouse application. When a picture of a person holding a box, the model identified the person but mislabeled the box as a laptop, as shown in Figure 3(a). When a standard dog image was tested, the model successfully identified the dog, as illustrated in Figure 3(b). This indicates that the pre-trained YOLOv8n.pt model, which was likely trained on general objects, failed to identify boxes for a warehouse application.

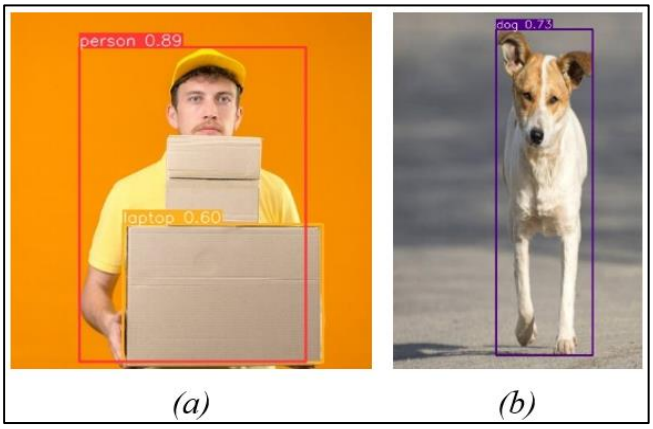

Fig 3 Predicted output Image for Approach 1

After 100 epochs of training for Approaches 2 and 3, the model's bounding box loss, classification loss, and distribution focal loss (DFL) continue to decrease as shown in Figure 4, indicating that the model has learned effectively. In addition, the validation values of bounding box loss, classification loss, and distribution focal loss for each epoch are plotted in Figure 5.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP1017

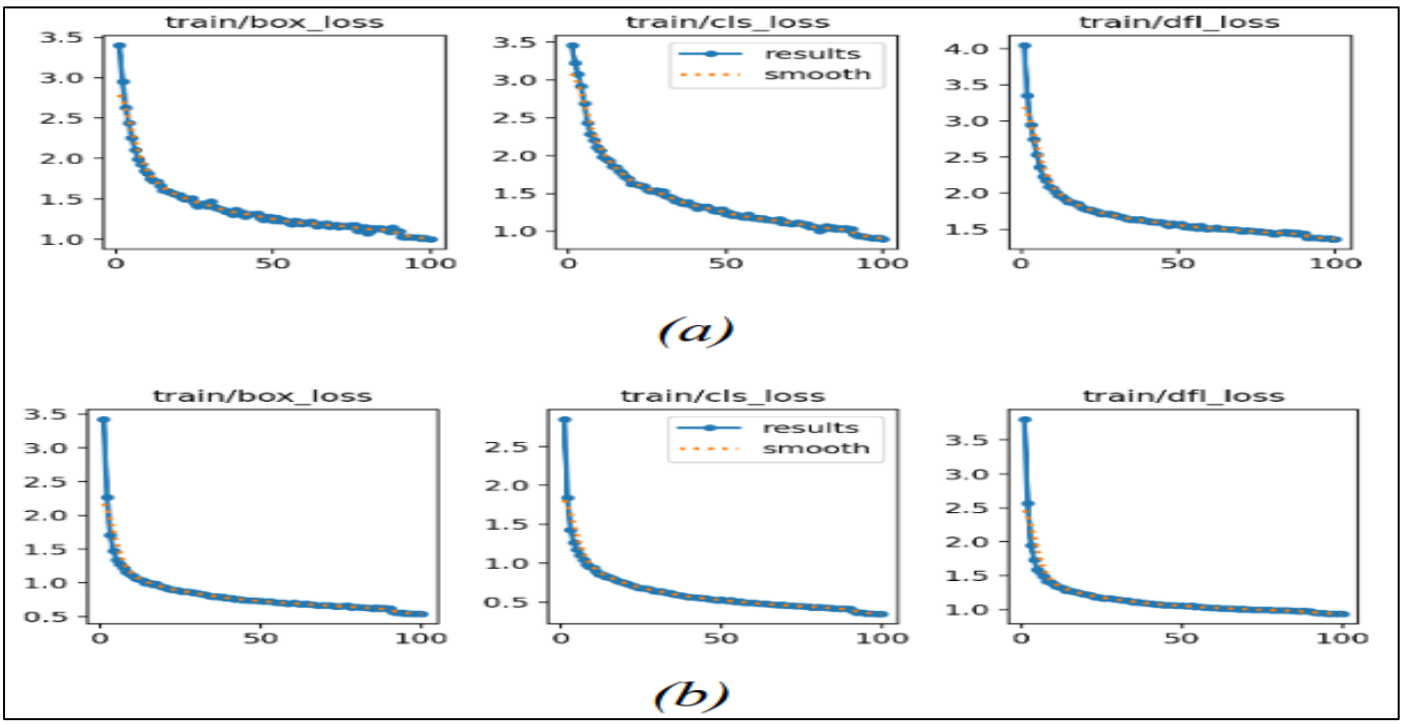

Fig 4 Bounding Box Loss, Classification Loss, and Training Distribution Focal Loss for Training (a) Approach 1, (b) Approach 2

For Approach 2, when predictions were made during inference, the model gave better results than Approach 1, see Figure 6 Although it correctly identified some boxes, others were misidentified or missed entirely. Although the Internet-sourced dataset improves the performance of the model, it lacks the accuracy required for warehouse environment, resulting in incorrect predictions during inference.

The model trained with a custom dataset (i.e., Approach 3) produced best results compared to previous approaches. For the sample images shown in Figure 7, the model accurately identifies all warehouse boxes.

When we compare the results of the image where a man is holding a box, Approach 2 failed but Approach 3 produced a detection with a smaller IOU see the Figure 8.

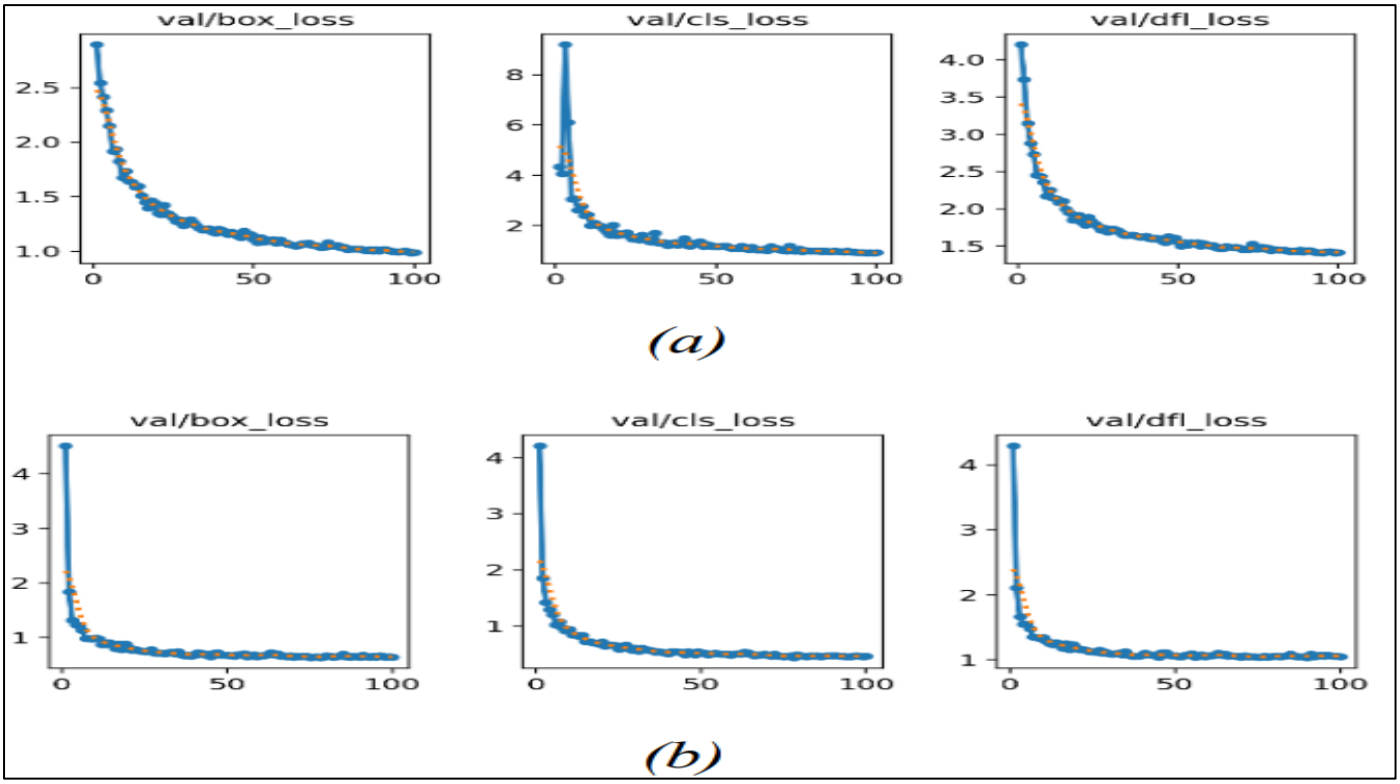

Fig 5 Bounding Box Loss, Classification Loss, and Training Distribution Focal Loss for validation (a) Approach 1, (b) Approach 2

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP1017

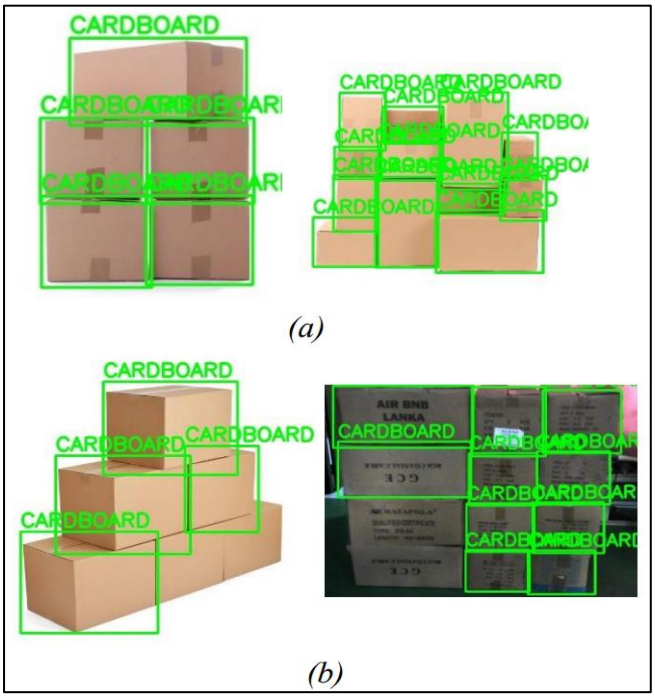

Fig 6 Predictions for Approach 2: (a)Successful Results, (b) Unsuccessful Results

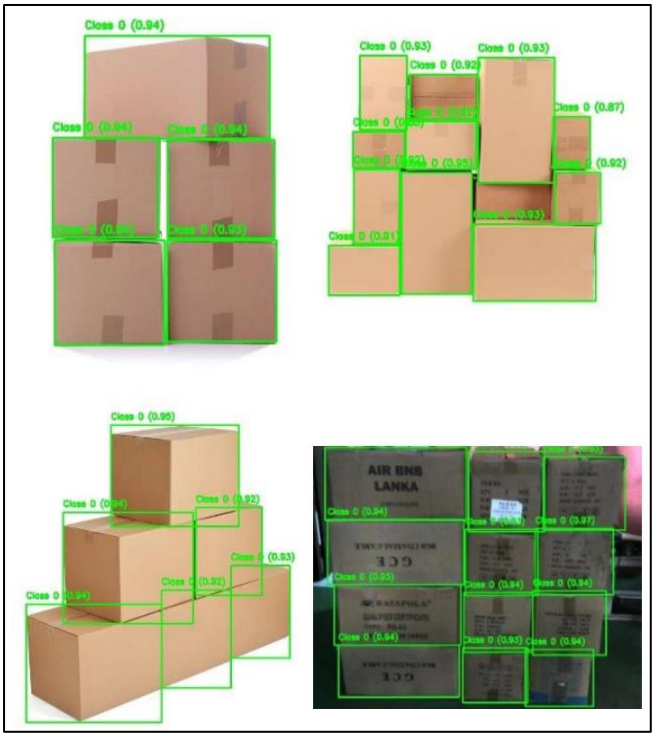

Fig 7 Prediction Output for Approach 3

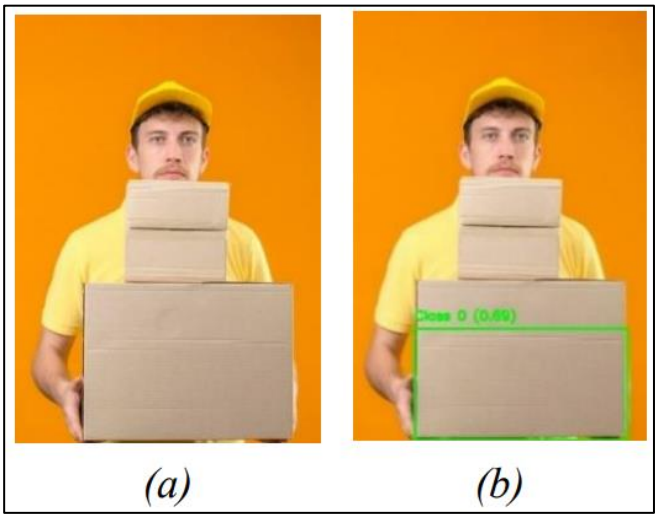

Fig 8 Predicted Image for the Scenario of a Person Holding a Box: (a) Approach 2, (b) Approach 3

After training the machine learning models, performance evaluation is crucial to determine the model's accuracy. Mean Average Precision (mAP) serves as a key performance indicator, helping developers and researchers gauge the effectiveness of their models [7]. Model accuracy was evaluated using precision (P) in Equation 1, recall (R) in Equation 2, and mean average precision (mAP) in Equation 3 as the chosen evaluation metrics [9].

Equation 1

$$P = \frac{TP}{TP + FP}$$

Equation 2

$$R = \frac{TP}{TP + FN}$$

Equation 3

$$mAP = \frac{\sum_{i=1}^{N} \int_0^1 P(R)dR}{N}$$

To evaluate the YOLOv8n-trained model, the output results were examined after validation. For both Approach 2 and 3, precision (P), recall (R) and mAP values as shown in Table 2.

Table 2 Validation Results for Approach 2 and 3

Evaluation metrics	Approach 2	Approach 3
P	0.707	0.939
R	0.599	0.885
mAP50	0.643	0.934
mAP50-95	0.45	0.821

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP1017

IV. CONCLUSION

This study demonstrates how the YOLOv8 model can be effectively used for warehouse box identification. We have demonstrated that YoloV8 performance can be improved using a training data set specific to the warehouse application under various lighting conditions. Although our best method shows a significant improvement, it is still not suitable for real applications in warehouses. We are continuing our research to improve the performance by application of fault-tolerant computing methods.

REFERENCES

[1]. M. van Geest, B. Tekinerdogan, and C. Catal, "Smart warehouses: Rationale, challenges and solution directions," Jan. 01, 2022, MDPI. doi: 10.3390/app12010219.

[2]. T. Xie and X. Yao, "Smart Logistics Warehouse Moving-Object Tracking Based on YOLOv5 and DeepSORT," Applied Sciences (Switzerland), vol. 13, no. 17, Sep. 2023, doi: 10.3390/app13179895.

[3]. G. Lavanya and S. D. Pande, "Enhancing Real-time Object Detection with YOLO Algorithm," EAI Endorsed Transactions on Internet of Things, vol. 10, 2024, doi: 10.4108/eetiot.4541.

[4]. M. Sohan, T. Sai Ram, and Ch. V. Rami Reddy, "A Review on YOLOv8 and Its Advancements," 2024, pp. 529–545. doi: 10.1007/978-981-99-7962-2_39.

[5]. M. Talib, A. H. Y. Al-Noori, and J. Suad, "YOLOv8-CAB: Improved YOLOv8 for Real-time Object Detection," Karbala International Journal of Modern Science, vol. 10, no. 1, pp. 56–68, 2024, doi: 10.33640/2405-609X.3339.

[6]. N. Ma, Y. Wu, Y. Bo, and H. Yan, "Chili Pepper Object Detection Method Based on Improved YOLOv8n," Plants, vol. 13, no. 17, p. 2402, Aug. 2024, doi: 10.3390/plants13172402.

[7]. P. Yennamaneni, S. Sabannawar, S. Naroju, and B. K. Depuru, "Improving Warehouse Efficiency Through Automated Counting of Pallets: YOLOv8-Powered Solutions," 2023. [Online]. Available: www.ijisrt.com

[8]. "Roboflow Universe: Open Source Computer Vision Community." Accessed: Sep. 02, 2024. [Online]. Available: https://universe.roboflow.com/

[9]. L. Wang et al., "Research on improved YOLOv8n based potato seedling detection in UAV remote sensing images," Front Plant Sci, vol. 15, 2024, doi: 10.3389/fpls.2024.1387350.