Volume 10, Issue 3, March – 2025 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

Intelligent Vision System for Real-Time Pallet Detection, Counting and Efficient Warehouse Management

Kunal G. Borase[1]; Sowmiya R[2]; Bharani Kumar Depuru[3]

[1;2;3]Aispry

Publication Date: 2025/03/28

Abstract: Nowadays, keeping track of inventory accurately is a big challenge in warehouses, especially when dealing with large-scale pallet production. Traditional methods like manual counting can lead to errors, such as incorrect counts, causing inefficiencies and delays in order fulfillment. These issues not only slow down operations but also increase costs and impact the entire supply chain.

This paper implements an high level solution deploying AI-powered system using deep learning and computer vision to accurately count stacked pallets in real-time, improving inventory management and operational efficiency to develop a reliable detection system, extensive data collection was carried out in warehouse settings capturing pallet images under different conditions the dataset was carefully labeled using polygon-based annotations via an open source annotation tool roboflow with extensive facilities for augmentations. To ensure precise object detection various data augmentation techniques such as shear, exposure, noise, blur, grayscale, horizontal flip, saturation, rotation and brightness adjustments were applied to improve model robustness against real-world variations

For real-time and high-accuracy pallet detection and counting , YOLO (You Only Look Once) object detection models like YOLOv8, YOLOv9, and YOLO11 were used for training and optimization. These models offered fast inference speeds, ensuring low latency while maintaining high detection precision. A comparative analysis of different YOLO versions provided insights into model performance, accuracy, and efficiency.

The optimized model was implemented in a warehouse setting and connected to a real-time monitoring system, enabling automatic pallet counting and reducing manual effort and errors. This research highlights the Switching from traditional inventory management to an AI-powered system, presenting how deep learning can enhance precision, minimize human mistakes, and cut operational expenses. By implementing an AI-driven vision system, Businesses can optimize supply chain processes, enhance order accuracy, and implement scalable warehouse automation.

Keywords: *Pallet Detection and Counting, Object Detection, Annotations, YOLOv8, YOLOv9, YOLO11, Deep Learning, Computer Vision, Warehouse Automation, Inventory Management.*

How to Cite: Kunal G. Borase; Sowmiya R; Bharani Kumar Depuru (2025). Intelligent Vision System for Real-Time Pallet Detection, Counting and Efficient Warehouse Management. *International Journal of Innovative Science and Research Technology*, 10(3), 1603-1611. https://doi.org/10.38124/ijisrt/25mar1269

I. INTRODUCTION

The increasing scale of warehouse operations presents a growing challenge in maintaining inventory accuracy. Manual and barcode-based pallet counting methods struggle to keep up with the high volume of pallet movements, leading to discrepancies in stock records, delays in order processing, and increased operational costs. Ensuring real-time and precise pallet detection is essential for improving inventory management, minimizing errors, and enhancing supply chain efficiency. This research focuses on leveraging artificial intelligence to automate pallet detection and counting[1][7], eliminating inefficiencies in traditional inventory tracking.

Unlike traditional methods, AI-based vision systems provide a scalable and automated solution for warehouse inventory management. The proposed system is designed to detect and count pallets accurately, even when they are stacked at varying heights and orientations. By integrating deep learning techniques, the system enables warehouses to achieve faster processing times, reduced manual intervention, and improved tracking accuracy. The implementation of YOLOv8[2][7], YOLOv9[3][8], and YOLO11[4][9] ensures a high level of precision and real-time inference capability, making it suitable for large-scale operations.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

To develop a robust detection system, a comprehensive dataset of warehouse pallet images was collected and annotated using polygon-based annotation in Roboflow[6]. This ensured high-quality training data, enabling the model to distinguish pallets in complex warehouse environments. Additionally, data augmentation methods, including brightness adjustments, scaling, rotation, shear transformations, exposure modifications, blur effects, grayscale conversion, and horizontal flipping, were applied to enhance model generalization, improving its ability to perform under diverse conditions.

A structured methodology CRISP-ML(Q)[5] was followed to build an efficient pallet detection system. The research involved training multiple YOLO models, evaluating their performance using key metrics such as mAP50 (Mean Average Precision at 50% IoU), mAP50-95 (Mean Average Precision across 50% to 95% IoU), precision, recall, and confusion matrix analysis, and selecting the best-performing model for deployment. By deploying the AI-powered vision system on warehouse cameras, real-time inventory tracking was achieved, reducing the dependency on manual stock verification and enhancing overall warehouse efficiency.

The study explores how deep learning-based object detection[8][9] can revolutionize inventory management by providing real-time tracking, increased accuracy, and cost-effectiveness. By implementing an AI-powered approach, businesses can minimize errors, improve operational workflow, and streamline warehouse automation, making this solution a valuable contribution to modern logistics and supply chain management.

The project methodology followed here is the open source CRISP-ML(Q)[5] methodology from 360DigiTMG (ak.1) [Fig 1], which stands for Cross Industry Standard Process for Machine Learning with Quality Assurance. This structured methodology ensures a systematic approach to problem identification, data preprocessing, model training, evaluation, and deployment. By following CRISP-ML(Q)[5], the project follows a well-defined lifecycle from data collection to real-world implementation, ensuring robust and efficient model performance.

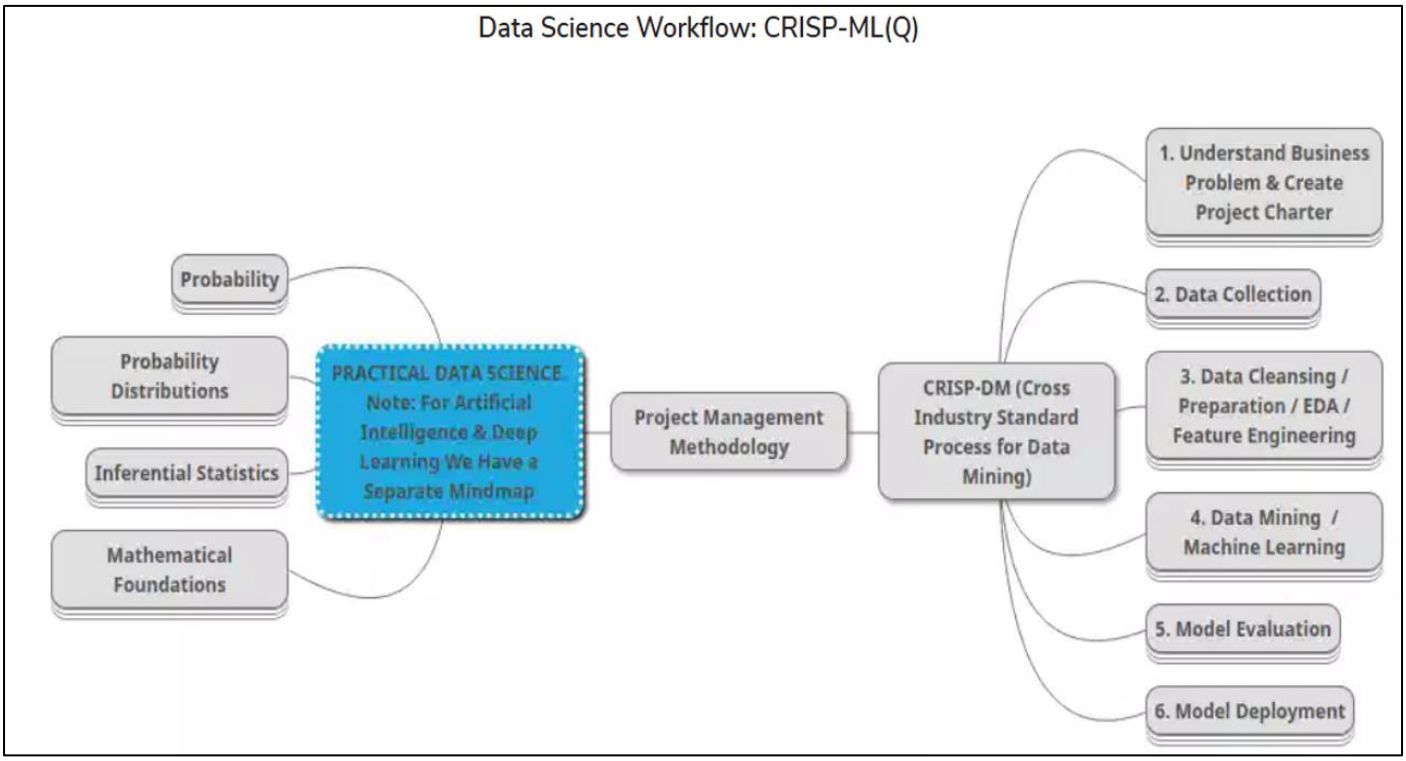


Fig 1 This Figure Depicts the CRISP-ML(Q) Architecture that We have followed for this Research Study.
(Source: Mind Map - 360DigiTMG)

II. METHODOLOGY AND TECHNOLOGY

➢ *Data Collection*

The dataset for this research was collected from a client's warehouse under varying lighting conditions to ensure robust model performance in real-world scenarios. A total of 2000+ images of stacked pallets were captured from a front-view perspective, covering different stacking patterns and occlusions. The dataset includes wooden and plastic pallets with different colors such as blue, yellow, and red to enhance model generalization. Additionally, images were sourced from secondary data sources, including open source platforms, further expanding the dataset. The collected images had varying dimensions, including 1200×765 and 225×225, ensuring diversity in scale and resolution. The structured dataset provides a strong foundation for training an AI-powered vision system capable of accurately detecting and counting[1][7] stacked pallets in different warehouse environments.

Below table [Fig 2] shows Data Description.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

Attribute	Description
Total Images	2000+ Original Images
Data Collection Source	Client's warehouse and Open source platforms
Data Type	Unstructured Dataset
Image Format	JPG, JPEG, PNG.
Image Type	RGB
Pallets Colors	Blue, Yellow, red, etc.
Image View	Front and side view perspective
Image Resolutions	1200×765, 225×225, etc.

Fig 2 Data Description

➢ *Data Annotations and Augmentations*

To ensure precise pallet detection, polygon-based annotation was carried out using Roboflow[6] shown in [Fig 3]. This method was chosen over traditional bounding boxes to accurately capture the irregular edges and varied orientations of pallets. Polygon-based annotation helped eliminate unnecessary background elements, enhancing the clarity and precision of object detection[8][9]. The dataset included a diverse range of pallets captured from different angles and under varying lighting conditions to ensure robustness in real-world scenarios.

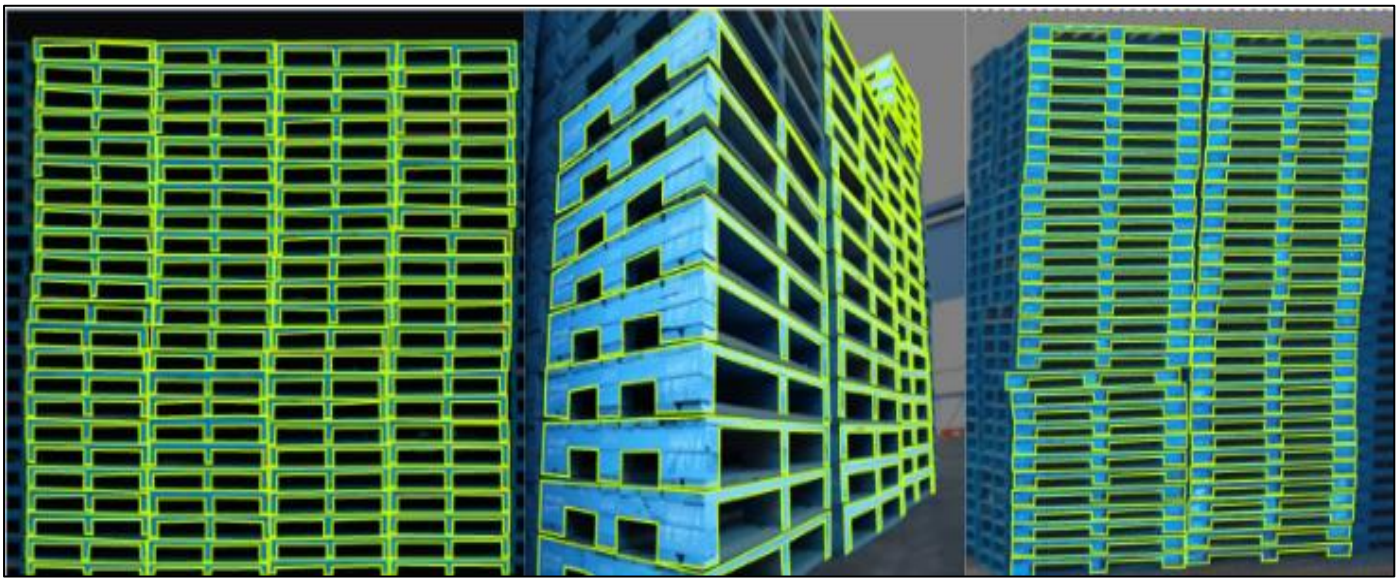


Fig 3 Polygon-based Annotated Images using RoboFlow

Before augmentation, the dataset consisted of 2000+ original images. To further improve the model's ability to detect pallets in challenging warehouse environments, data augmentation was applied. Various augmentation techniques were introduced to replicate real-world variations, including shear transformations, brightness adjustments, rotation,

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

exposure modifications, blur effects, grayscale conversion, and horizontal flipping. These augmentations allowed the model to generalize better across different warehouse conditions, reducing the risk of detection errors due to variations in pallet stacking, lighting, or partial occlusions.

To maintain consistency in model input, preprocessing steps were also applied. All images were auto-oriented to ensure correct alignment, and their dimensions were standardized to 640×640 pixels for optimal training performance.

Following annotation and augmentation, the dataset expanded significantly to 22,220 images, providing a more diverse and balanced training set. This expansion helped improve the model's accuracy in detecting and counting pallets[1][7], making it more adaptable to different warehouse settings.

To systematically enhance model robustness, four different versions of augmentation were applied, each version generated multiple images and added extensive diversity onto the original image, which helped in introducing specific transformations to improve generalization. [Fig 4] presents a detailed breakdown of the augmentation strategies applied in this study.

Versions	Augmentations	Description
Version 1	Shear (±10°), Brightness (±20°), Noise (1.0%)	Simulates slight perspective distortions, varying light conditions, and minor noise.
Version 2	Rotation (±15°), Exposure (±10%), Noise (1.0%)	Introduces rotation variations, exposure adjustments, and noise to mimic real-world inconsistencies.
Version 3	Blur (1.5 px), Saturation (30), Noise (1.0%)	Applies blurring for slight defocus, alters saturation for color intensity, and adds noise.
Version 4	Grayscale (100%), Hue (±15°), Flip (Horizontal)	Converts images to grayscale, adjusts hue for color shifts, and applies horizontal flipping to increase diversity.

Fig 4 Augmentation Strategies Used in the Study

These augmentations played a crucial role in simulating real warehouse conditions, by ensuring that the dataset included a variety of augmented images, the system was trained to handle variability in lighting, perspective changes, and partial occlusions, leading to a more resilient and accurate detection model.

Moving forward, the dataset was subdivided into three partitions: training, testing, and validation. The split ratio was set as 90% for training, 5% for testing, and 5% for validation. Additionally, 50–60 real world images which were intentionally not a part of the training dataset are kept as unseen data for the model building. These images were later used to assess the model's ability to generalize and improve detection performance on entirely new samples.

➢ *Model Architecture*

The pallet detection system follows a structured architecture that integrates multiple stages, including data collection, preprocessing, model training, evaluation, and deployment[5]. This approach ensures an end-to-end workflow, optimizing both model accuracy and real-time performance for warehouse environments. Below shows our real world project architecture which describes the business flow followed by gathering data and applying extensive preprocessing ,proceeded with building and evaluating the model which is finally deployed on a cloud environment. The high-level model architecture is illustrated in [Fig 5].

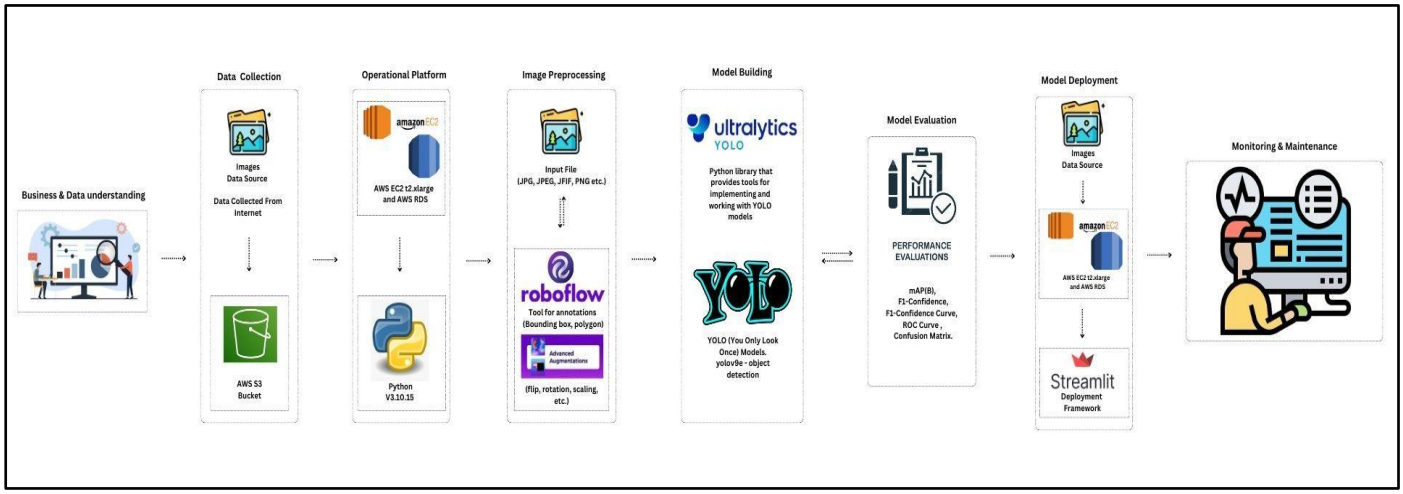


Fig 5 End-to-End High Level Architecture of the Pallet Detection System

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

This architecture demonstrates the end-to-end pipeline, from image acquisition to model evaluation and deployment, ensuring that the system is scalable and efficient for real-world warehouse applications.

➢ *Model Building*

To develop an accurate and efficient pallet detection system, multiple YOLO (You Only Look Once) models were trained and evaluated to identify the most suitable architecture for warehouse environments. The models are trained on multiple variants of different YOLO versions which includes YOLOv8m(medium), YOLOv8n(nano), YOLOv8l(large), YOLO11m(medium), and YOLOv9c (Compact), each tested under different epoch settings to assess their performance in terms of detection accuracy and computational efficiency. The objective was to select a model capable of accurately detecting pallets in all possible scenarios of stacked positions with better accuracy. By analyzing the performance of each model, the most effective configuration was determined to ensure precise and real-time pallet counting for improved warehouse automation.

- *YOLOv8n* –

The nano (n) version of YOLOv8[2][7] is optimized for fast inference with minimal computational requirements. It consists of 3.2Million parameters with FLOPs (Floating Point Operations per Second) as 8.7 Billion.It is a lightweight model suitable for real-time applications where processing speed is crucial. In this project, it was tested to determine whether a compact architecture could still achieve accurate pallet detection while maintaining high processing speeds, making it viable for scenarios where rapid inventory updates are required.

- *YOLOv8m* –

This model belongs to the YOLOv8[2][7] family and is a medium-sized variant, balancing detection accuracy and inference speed. It consists of 25.9Million parameters with FLOPs (Floating Point Operations per Second) as 78.9Billion. It is designed for applications requiring moderate computational power while maintaining reliable object detection[8][9] performance. In pallet detection, it was used to analyze how well a mid-sized YOLO model adapts to varying lighting conditions and stacking complexities in warehouse environments.

- *YOLOv8l* –

The large (l) version of YOLOv8[2][7] provides higher accuracy due to its increased network depth and complexity. It consists of 43.7Million parameters with FLOPs (Floating Point Operations per Second) as 165.2Billion. This model was trained to evaluate its ability to handle occlusions, densely stacked pallets, and challenging warehouse conditions. While more computationally demanding, it offers superior detection precision on small objects , making it suitable for environments requiring high accuracy over speed.

- *YOLOv9c* –

This variant is part of the YOLOv9[3][8] architecture, which includes advancements in detection precision and model efficiency. It consists of 25.5Million parameters with FLOPs (Floating Point Operations per Second) as 102.8Billion. The c version of YOLOv9[3][8] was chosen to evaluate its ability to identify pallets in diverse warehouse conditions, including low lighting and partial occlusion scenarios. By incorporating architectural improvements, this model was tested for enhanced object tracking and stability in real-world applications.

- *YOLO11m* –

An upgraded version of YOLO designed to improve detection accuracy and feature extraction compared to previous models. It consists of 20.1Million parameters with FLOPs (Floating Point Operations per Second) as 68 Billion. The medium (m) version of YOLO11[4][9] was selected to assess how well a more advanced model performs in differentiating pallets in complex backgrounds. It was particularly useful for detecting pallets with similar colors, shapes, and stacked orientations, improving overall model robustness.

➢ *Model Evaluation*

In this paper different YOLO models from Ultralytics were trained and tested,the selection process followed Decision Analysis and Resolution (DAR) [Fig.4], where multiple YOLO models were compared based on accuracy, resource usage, ease of use, system requirements, and scalability.

The evaluation involved six different YOLO models: YOLOv8m (70 epochs), YOLOv8m (100 epochs), YOLOv8l (100 epochs), YOLOv9c (100 epochs), YOLO11m (70 epochs), and YOLO11m (100 epochs). Each model was assessed for its performance in pallet detection under different warehouse conditions, including variations in lighting, stacking patterns, and occlusions. The comparison was based on key performance indicators such as mAP50, mAP50-95, precision, recall, confusion matrix analysis, and system resource consumption.

The confusion matrix was utilized to analyze the model's performance in distinguishing pallets from background noise and other non-relevant objects. It provided a detailed breakdown of true positives (correct detections), false positives (incorrect detections), false negatives (missed detections), and true negatives (correct non-detections), ensuring a comprehensive assessment of detection accuracy.

The results demonstrated that YOLOv8L (100 epochs) outperformed the other models in terms of accuracy while maintaining a reasonable computational footprint. The model achieved an mAP50 of 0.994 (99.4%), an mAP50-95 of 0.838 (83.8%), a precision of 0.98 (98.9%), and a recall of 0.986 (98.6%), making it the most reliable model for pallet detection and counting[1][7]. The mAP50 score, which measures detection accuracy at a 50% Intersection over Union (IoU) threshold, indicated that YOLOv8L (100 epochs) provided the highest precision among all tested models. The mAP50-95 score, which evaluates accuracy across multiple IoU thresholds, further confirmed the robustness of this model in detecting pallets under varying

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

conditions such as different lighting, stacking orientations, and occlusions.

In addition to accuracy, the computational efficiency of each model was analyzed. While YOLOv8l (100 epochs) required moderate GPU resources where g4dn AWS EC2 Ubuntu Scalable instance was used, it provided the most optimal balance between high detection accuracy and efficient processing, making it suitable for deployment in real-time warehouse environments. Compared to other models, it maintained high detection performance while ensuring feasibility for large-scale warehouse automation.

Below table [Fig 6] presents a detailed comparison of the YOLO models, summarizing their performance across multiple evaluation criteria. Based on these insights, YOLOv8L (100 epochs) was chosen as the final model for deployment, providing an optimal trade-off between accuracy, speed, and system efficiency for automated pallet detection in warehouses.

Criteria	YOLOv8m (70 epochs)	YOLOv8n (100 epochs)	YOLOv8l (100 epochs)	YOLOv9c (100 epochs)	YOLO11m (70 epochs)	YOLO11m (100 epochs)	Decision
Performance & Resource Utilization	Moderate compared to YOLOv8l & YOLOv11m	Good, improved over 70 epochs but more compute needed	High accuracy with slightly higher compute cost	High accuracy with improved efficiency over YOLOv8l, moderate compute cost.	Similar to YOLOv8m (100 epochs), better than 70 epochs	Higher compute requirements, slightly better than YOLOv8l	**YOLOv8L (100 epochs)** chosen for best accuracy-performance balance
Ease of Use	Easy to train, moderate results, medium training and inference speed.	Very Easy to train, no stable results, fastest training and inference speed.	Harder to train, stable results with slower training and inference speed.	Easy to train, less stable results, fast training and inference speed.	Harder to train, moderate stable results, slower training and inference speed.	Requires longer training time, higher complexity	**YOLOv8L (100 epochs)** is manageable with high accuracy
Accuracy and Quality	**mAP50:** 0.963, **mAP50-95:** 0.704, **Precision:** 0.926, **Recall:** 0.911	**mAP50:** 0.985 **mAP50-95:** 0.774, **Precision:** 0.961, **Recall:** 0.952	**mAP50:** 0.994, **mAP50-95:** 0.838, **Precision:** 0.989, **Recall:** 0.986	**mAP50:** 0.986 **mAP50-95:** 0.819 **Precision:** 0.967 **Recall:** 0.986	**mAP50:** 0.987, **mAP50-95:** 0.779, **Precision:** 0.961, **Recall:** 0.957	**mAP50:** 0.992, **mAP50-95:** 0.8, **Precision:** 0.974, **Recall:** 0.969	**YOLOv8L (100 epochs)** has the highest accuracy
System Requirements	Low GPU requirements	Slightly higher GPU required	Moderate GPU required	Moderate GPU required,	Highest GPU required	Highest GPU required	**YOLOv8L (100 epochs)** offers the best accuracy while remaining feasible
Scalability	Suitable for Edge & Cloud	Good scalability, efficient for deployment	Deployable on cloud & optimized for warehouse applications	Deployable on cloud, optimized for both edge and large-scale applications.	Good for deployment	Less scalable, due to high resource requirements	**YOLOv8L (100 epochs)** chosen for best trade-off between scalability & accuracy

Fig 6 Model Selection and Performance Evaluation using Decision Analysis and Resolution (DAR)

This process ensures that the chosen Ultralytics YOLO model meets the needs for automated inventory tracking, improved accuracy, and real-time use, making it a scalable solution for warehouse management.

➢ *Model Hyperparameter Tuning*

After selecting YOLOv8L (100 epochs) as the best-performing model, we adjusted its hyperparameters to improve accuracy and efficiency. The goal was to make the model better at detecting pallets with high precision while keeping it fast and efficient for real-time use. The model worked well in most cases, but it had some difficulties. When there were more than 150 or 200 pallets stacked together or when the image was taken from far away, the model struggled to detect and count[1][7] all the pallets correctly. This showed that while the model performs well in general, it needs further improvements to handle large stacks of pallets in big warehouses.

To further increase the performance of the selected YOLOv8L (100 epochs) model, hyperparameter tuning was conducted to enhance object detection accuracy. we experimented with key hyperparameters like initial learning rate,final learning rate, seed ,warmup epochs etc was added and further trained with 50 more epochs by taking the best.pt which is the outcome of YOLOV8L by enabling pretrained = True parameter, accuracy was improved and inference speed was moderate. The following hyperparameters in [Fig 7] were selected based on experimental analysis and best practices.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

Model	Epochs	Batch Size	Hyperparameters
YOLOV8L	50 epochs	16	seed=100, lr0=0.002, lrf=0.1, warmup_epochs=5.0, verbose=True, plots = True, pretrained =True

Fig 7 Hyperparameter Tuning for the best Model

After adjusting the hyperparameters, our model showed a significant improvement in detection and counting accuracy. The results are shown in the following table [Fig 8].

Metrics	Performance
mAP50	Measures AP at **50% IoU threshold 0.986 (98.6%).**
mAP50-95	Measures AP across IoU thresholds from **50% to 95% 0.823 (82.3%).**
Precision	Measures how often detected pallets are correct **0.971 (97.1%).**
Recall	Measures how many actual pallets were detected **0.965 (96.5%)**

Fig 8 Model Performance Metrics After Hyperparameter Tuning

With these hyperparameter tuning, the model can now accurately detect pallets and count, them correctly. The 97.1% precision means the model makes very few mistakes in identifying pallets, while the 96.5% recall shows that it successfully detects most pallets present in the images. The 98.6% mAP@50 confirms high detection accuracy at a 50% overlap, and the 82.3% mAP@50-95 shows that the model performs well.

III. MODEL DEPLOYMENT

After adjusting the hyperparameters, we selected YOLOv8L as the best model for deployment. To make it easy to use, we built a Streamlit web app. Streamlit[10] is a simple python framework that helps create interactive applications for AI/ML applications where users can upload images and see detection results instantly.

The web app allows users to upload an image, and the model will detect and count pallets in real time. It provides a clear and easy-to-use interface, making it suitable for different deployment environments, whether on a local computer or a cloud server. The landing page of the web app is shown in the figure below [Fig 9].

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

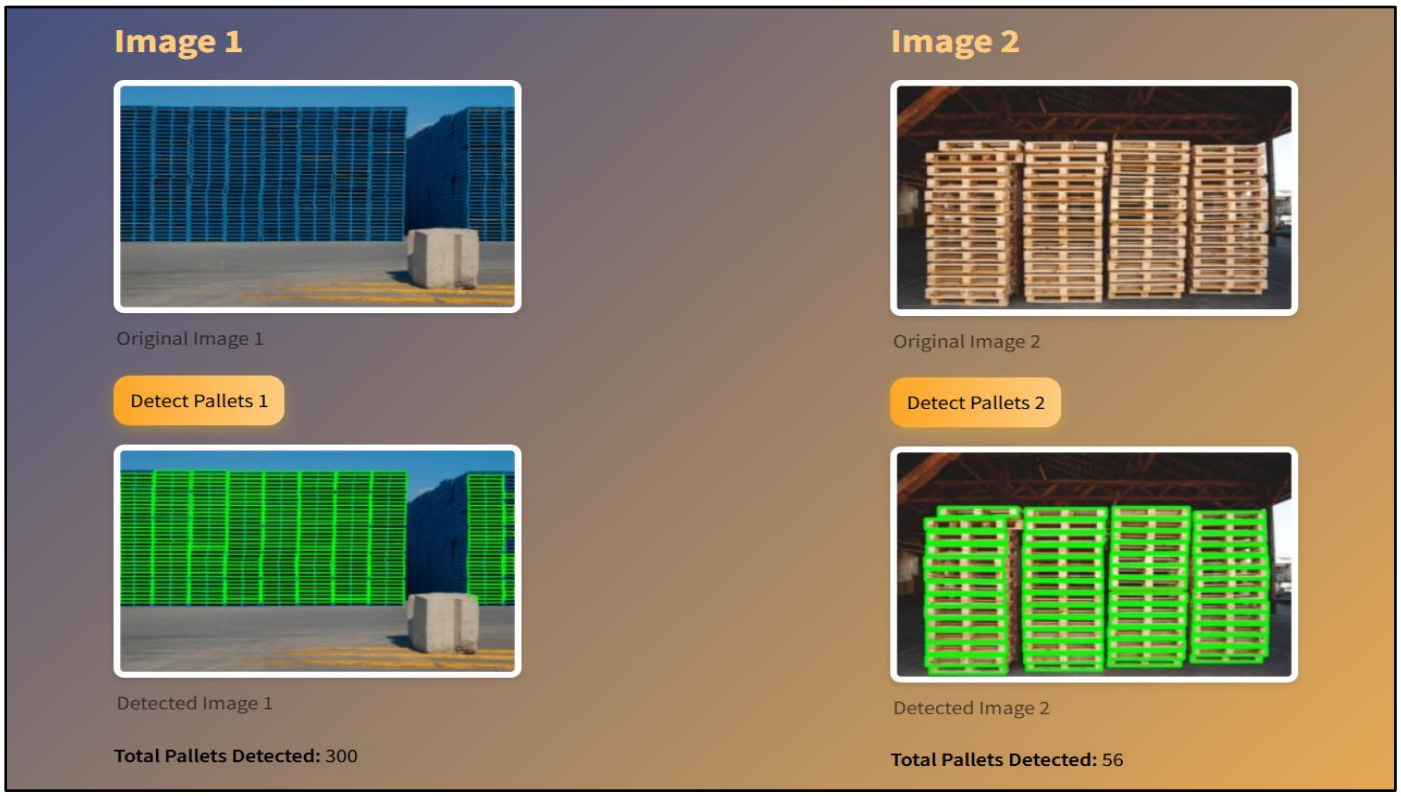
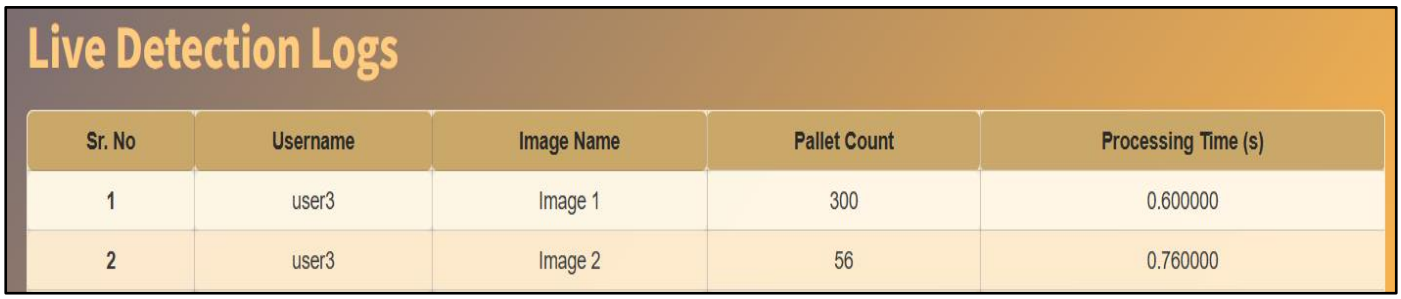


Fig 9 Streamlit UI Detection and Counting Results.

To track and evaluate the performance of the deployed AI model, a Live Detection Logs table has been added to the system [Fig 10]. This table helps in monitoring pallet detection results in real time, ensuring accuracy and efficiency in warehouse operations.



Sr. No	Username	Image Name	Pallet Count	Processing Time (s)
1	user3	Image 1	300	0.600000
2	user3	Image 2	56	0.760000

Fig 10 Real-Time Detection and Counting Results

- *This Table [Fig. 10] Includes the Following Details:*

✓ **Sr. No:** A unique number for each detection entry.
✓ **Username:** The user who uploaded the image for detection.
✓ **Image Name:** The name of the image processed by the model.
✓ **Pallet Count:** The total number of pallets detected in the image.
✓ **Processing Time (s):** The time taken by the model to analyze the image and provide results.

IV. CONCLUSION

The evaluation of YOLOv8L for pallet detection and counting has shown strong performance, making it a good choice for real-world industrial use. The model accurately detects and counts pallets, even with differences in size, color, and arrangement (i.e stacked pallets). Fine-tuning helped achieve high accuracy, with 97.1% precision, 96.5% recall,a mean Average Precision (mAP@50) of 98.6% and (mAP@50-95) of 82.3%. These improvements help with better storage management and overall efficiency.

During testing, the model worked well in various scenarios, including stacked pallets. However, challenges arose when detecting large numbers of pallets (150-200) from long distances, showing areas for improvement. Enhancing the dataset and refining the model could further improve accuracy.

Using Streamlit for deployment provides an easy-to-use web interface, enabling real-time pallet detection with minimal computing power. This makes the system accessible for warehouse automation and stock management.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/25mar1269

Overall, this project demonstrates the potential of YOLOv8L for automating pallet detection and counting. These advancements can also be applied to other industries needing object detection and tracking, helping to make warehouse operations and supply chain management more efficient and automated.

FUTURE SCOPE

In the future, we can use IoT (Internet of Things) sensors to improve pallet tracking and make inventory updates more automatic. The model can also be deployed on small devices like tablets or mobile phones by converting this AI application into mobile application, so workers can easily detect and count pallets on the go. Additionally, connecting this system with Warehouse Management Systems (WMS) will help update stock levels in real time, making inventory management more efficient.

ACKNOWLEDGMENTS

We sincerely thank 360DigiTMG for providing us with the opportunity to work on this project. We also appreciate the guidance and support of our partners throughout this research. Additionally, we acknowledge the use of the CRISP-ML(Q) framework and ML Workflow, which are openly available on the official 360DigiTMG website and used with their explicit consent.

REFERENCES

[1]. Ishwari Sidwadkar, Pranjali Bandgar, Akshada Khadke, Tahesin Pathan, Prof S. R. Bhujbal People Counting, Capturing Image Using YOLO Deep Learning Algorithm https://www.jetir.org/papers/JETIR2404784

[2]. Mupparaju Sohan1, Thotakura SaiRam, and Ch. Venkata RamiReddy. A Review on YOLOv8 and Its Advancements. DOI:10.1007/978-981-99-7962-2_39

[3]. Ning Kang, Fangyu Ma, Wenkang Wan, Daihan Wang, Hua Yao, Kai Sheng. Improved YOLOv9-Based Objects Detection in Adverse Weather Conditions for Autonomous driving. https://doi.org/10.1016/j.ifacol.2024.11.155

[4]. Rahima Khanam and MuhammadHussain ,YOLO11: An Overview of the Key Architectural Enhancements. https://arxiv.org/abs/2410.17725

[5]. Stefan Studer , Thanh Binh Bui , Christian Drescher , Alexander Hanuschkin , Ludwig Winkler , Steven Peters and Klaus-Robert Müller Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology . https://doi.org/10.3390/make3020020

[6]. Prof. P. D. Kale, Comparative Analysis of Image Annotation Tools: LabelImg, VGG Annotator, Label Studio, And Roboflow", International Journal of Emerging Technologies and Innovative Research. https://www.jetir.org/papers/JETIR2405D59

[7]. Praneeth Yennamaneni; Vickram R; Samyak Sabannawar; SreePriya Naroju; Bharani Kumar Depuru ,Improving Warehouse Efficiency Through Automated Counting of Pallets: YOLOv8-Powered Solutions. https://doi.org/10.5281/zenodo.10276587

[8]. Fatma Betül Kara ArdaÇ; Pakize Erdogmus. Car Object Detection: Comparative Analysis of YOLOv9 and YOLOv10 Models. DOI: 10.1109/ASYU62119.2024.10756955

[9]. Mingming Zhang,Shutong Ye,Shengyu Zhao,Wei Wang andChao Xie. Pear Object Detection in Complex Orchard Environment Based on Improved YOLO11. https://doi.org/10.3390/sym17020255.

[10]. Himangi Dani, Pooja Bhople, Hariom Waghmare, Kartik Munginwar, Prof. Ankush Patil Review on Frameworks Used for Deployment of Machine Learning Model https://doi.org/10.22214/ijraset.2022.40222