International Journal of Electrical, Electronics and Computer Systems

ISSN: 2347-2820

Volume 15 Issue 1s (2026) | Pages: 150-157

AI Based Product Object Detection & Counting, Sorting

Nilima Patil[1], Nikhil Ahire[2], Ashutosh Shinde[3], Vaishnavi Gunjote[4], Vishal Sonawane[5]

[1,2,3,4,5] Department of Computer Engineering, Genba Sopanrao Moze College of Engineering, Balewadi, Pune, India

Peer Review Information

***Type**: Article*
***Received**: 18 February 2026*
***Revised**: 15 March 2026*
***Accepted**: 12 April 2026*
***Published**: 21 May 2026*

Abstract

Object detection plays a crucial role in modern automation systems, particularly in areas such as retail inventor management, warehouse monitoring, and product categorization. Manual counting and classification of objects is labor- intensive, time consuming, and prone to human error. This project presents a YOLOv8-based real time object detection and classification system designed to detect, count, and categorize products into predefined classes such as Grocery, Medical, and Household items. The workflow includes dataset preparation, annotation, model training using YOLOv8, and post- processing steps for object counting and grouping. The YOLOv8 model is optimized for high- speed inference while retaining accuracy through fine tuning and anchor-free detection capabilities. Performance evaluation metrics such as MAP (Mean Average Precision), precision, recall, and inference time are used to assess model effectiveness. To enhance usability, the system provides real-time visual feedback with bounding boxes, labels, and product counts. Future improvements include integrating barcode recognition, implementing multi-camera tracking, and incorporating predictive analytics for demand forecasting.

Keywords: Artificial Intelligence; Computer Vision; Object Detection; YOLOv8; Deep Learning; Convolutional Neural Network.

How to Cite This Article

Patil, N., Ahire, N., Shinde, P., Gunjote, V., Sonawane, V. (2026). AI-Based Product Object Detection & Counting, Sorting. *International Journal of Electrical, Electronics and Computer Systems*, 15(1s), 150-157.

Introduction

Object detection has emerged as one of the most significant research areas in computer vision and artificial intelligence. It enables machines to automatically identify and localize multiple objects within images or video streams. With the rapid growth of retail, logistics, and industrial automation, the need for accurate and real-time object detection systems has increased considerably. Traditional methods of product counting and sorting rely heavily on manual labor or barcode- based systems, which are time-consuming and prone to human error. Recent advancements in deep learning, particularly Convolutional Neural Networks (CNNs), have significantly improved detection accuracy SSD, and YOLO have revolutionized the field by providing efficient detection frameworks. Among these, the YOLO (You Only Look Once) family of models is widelyrecognized for its real-time performance and high accuracy. The latest version, YOLOv8, introduces anchor-free detection and improved architecture for better speed and precision. These advancements make it suitable for practical applications such as inventory monitoring, product categorization, and automated sorting systems. This survey paper reviews existing object detection techniques, analyzes their strengths and limitations, and highlights the need for integrated systems capable of detection, counting, and categorization. The study also discusses current challenges and future research directions in real-time vision-based automation systems.

Motivation and Problem Statement:

The rapid growth of retail and warehouse industries has increased the need for automated inventory management systems. Traditional methods such as manual counting and barcode scanning are time-consuming and prone to errors. Although deep learning models have improved visual recognition, many systems focus only on detection and not on integrated counting and sorting. Real-world challenges such as occlusion and lighting variation reduce accuracy. There is a need for a real-time object detection and counting framework that works reliably in practical environments. The motivation of this work is to develop an AI-based intelligent solution that reduces human effort and improves operational efficiency. The primary objective of this project is to develop an AI-based real-time object detection system using the YOLOv8 model. The system aims to detect, count, and categorize products from images or live video streams. It focuses on improving accuracy and reducing human effort in inventory management. The project also aims to provide a scalable solution for industrial and retail applications.

Literature Review

Object detection and recognition in retail and grocery environments have been widely explored using deep learning and computer vision techniques. Prabu Selvam et al. [1] proposed a batch normalization-free rigorous feature flow neural network for grocery product recognition, utilizing a three-stage pipeline with YOLOv5 and a DDF refinement layer to enhance detection performance, especially for overlapping bounding boxes. Although the model improved detection accuracy, it still faced challenges in identifying small or occluded objects in crowded environments.

Keong-Hun Choi and Jong-Eun Ha [2] introduced an object detection method based on reinforcement learning that uses only image-level labels and object count information. While this approach reduces the dependency on detailed annotations, it lacks the ability to detect object categories and shows reduced performance when applied to complex datasets. Ji-Ye Jeon et al. [3] developed a retail object classification system using multiple cameras for vision-based unmanned kiosks. Their CNN-based approach improved detection accuracy through multiple viewpoints; however, the system's performance depends heavily on camera placement and degrades under occlusion or poor lighting conditions.

Min Qiao et al. [4] proposed a salient object detection method using a double-branch network combined with an Edge Profile Enhancement Module to handle complex-shaped objects. Despite achieving high accuracy, the model struggles in cluttered scenes and low-contrast backgrounds, limiting its real-world applicability. Ruben Sagues-Tanco et al. [5] focused on generating synthetic datasets for kitchen object segmentation in deep learning applications. Although the synthetic data provided competitive results, the performance remained slightly lower compared to handcrafted datasets, indicating limitations in dataset realism and generalization.

From the reviewed literature, a common gap can be observed: most existing systems either focus on improving detection accuracy, reducing annotation requirements, or enhancing dataset generation independently. However, very few approaches effectively combine real-time object detection, robustness in cluttered environments, lightweight deployment, and high accuracy within a single system. The proposed system aims to address this gap by integrating an efficient deep learning-based detection model capable of handling occlusion, varying lighting conditions, and real-time performance, making it suitable for practical retail and grocery applications.

System Architecture

The proposed AI-powered object detection system is designed to automatically detect, count, and categorize products in real time using advanced computer vision and deep learning techniques. The system is based on the YOLOv8 (You Only Look Once) algorithm, which

enables fast and accurate object detection by processing images or live camera feeds efficiently. It can identify multiple objects simultaneously and classify them into predefined categories such as Grocery, Medical, and Household, ensuring high performance in dynamic environments. The overall system architecture follows a structured and pipeline consisting of multiple stages.

Fig 1. *AI-Based Product Object Detection, Counting, and Sorting System Workflow and Real-Time Monitoring Dashboard*

Input Stage

The input stage is responsible for capturing data in the form of images or live video streams using a camera or uploaded files. This stage acts as the starting point of the system where raw visual data is collected for further processing. The quality of input data directly affects the accuracy of detection, making this stage crucial for system performance.

Preprocessing Stage

The preprocessing stage enhances the input data to make it suitable for model processing. It includes operations such as image resizing, noise reduction, normalization, and contrast adjustment. These techniques help in improving image quality and ensuring consistency, which ultimately leads to better detection accuracy.

Feature Extraction Stage

In this stage, important features are extracted from the processed images using deep learning techniques. The YOLOv8 model automatically identifies patterns, shapes, and object characteristics by analyzing different regions of the image. This step is essential for understanding the content of the image and preparing it for classification.

Classification and Detection Stage

The classification stage uses the YOLOv8 algorithm to detect objects and assign them to predefined categories such as Grocery, Medical, and Household. It generates bounding boxes around detected objects along with class labels and confidence scores. The model is capable of detecting multiple objects in a single frame, making it highly efficient for real-time applications.

Output Stage

The output stage presents the final results to the user in a clear and understandable format. It displays detected objects with bounding boxes, labels, and counts. This information helps in reducing manual effort, improving accuracy, and enhancing automation in inventory management systems.

Hardware and software requirement

Hardware Requirements

The hardware components play a crucial role in the efficient functioning of the AI-powered object detection system. Since the system performs real-time image processing and deep learning-based object detection using YOLOv8, it requires sufficient computational resources for smooth execution.

RAM: Minimum 8 GB RAM is required to handle image processing and model inference efficiently.

Processor: Intel i5 processor or higher is recommended for faster computation and smooth system performance.

Algorithm

- Hard Disk: At least 80 GB storage is required to store datasets, trained models, and system files.
- GPU (Optional but Recommended): A dedicated GPU improves the speed of model training and real-time detection.
- Display Monitor: Required for visualizing detection results such as bounding boxes and object labels.
- Keyboard and Mouse: Used for system interaction and control.

In summary, these hardware components ensure that the system performs efficiently, processes data in real time, and delivers accurate object detection results suitable for practical applications.

Software Requirements

The development environment and software dependencies required for building and running the system are as follows:

- Operating System: Windows, Linux, or macOS for development and execution of the system.
- Programming Language: Python is used due to its simplicity and strong support for machine learning and computer vision.
- Development Environment: Spyder IDE is used for coding, debugging, and running the application.
- Database: SQLite is used for storing detected object data and system records.
- Deep Learning Model: YOLOv8 is used for real-time object detection and classification.
- Computer Vision Library: OpenCV is used for image processing and handling camera input.
- Numerical Library: NumPy is used for mathematical operations and array processing.
- Visualization Library: Matplotlib is used for plotting graphs and visualizing results.

The system integrates Python with libraries such as YOLOv8, OpenCV, NumPy, and Matplotlib to perform efficient object detection and data processing. SQLite is used for lightweight data storage, while Spyder provides a convenient development environment. This combination ensures a robust, scalable, and efficient system suitable for real-time applications.

The system uses Python as the programming language with development tools like PyCharm or VS Code. It integrates frameworks such as TensorFlow or PyTorch and libraries like OpenCV and MediaPipe for gesture recognition. Text-to-speech tools like pyttsx3 or gTTS are used to provide audio output for better user interaction.

The proposed system uses the YOLOv8 (You Only Look

Once version 8) algorithm for real-time object detection and classification. YOLOv8 is a state-of-the-art deep learning model that processes the entire image in a single pass, making it highly efficient and suitable for real-time applications such as retail product detection. The algorithm is designed to detect multiple objects simultaneously while maintaining high accuracy and speed.

The algorithm proceeds as follows:

1.Input Acquisition

Capture input data in the form of images or live video streams using a camera or uploaded files. The system continuously receives frames in real time, which serve as input for object detection. The quality and resolution of input images directly impact detection performance.

2.Preprocessing

The captured input is preprocessed to improve quality and consistency. This includes resizing images to a fixed dimension (e.g., 640×640), normalization of pixel values, and noise reduction. Preprocessing ensures that the input data is suitable for the YOLOv8 model and improves detection accuracy.

3.Feature Extraction (Backbone Network)

The preprocessed image is passed through the backbone network of YOLOv8, which extracts important features such as edges, textures, and object patterns. This stage uses convolutional neural networks (CNNs) to learn hierarchical features from the image.

4.Neck Processing (Feature Aggregation)

In this stage, the model combines features from different layers using techniques like PAN (Path Aggregation Network) to improve detection of objects at multiple scales. This helps the system detect both small and large objects effectively.

5.Detection Head (Prediction Stage)

The detection head predicts bounding boxes, object classes, and confidence scores. YOLOv8 divides the image into grids and identifies objects within those regions. It generates multiple predictions for each object in a single forward pass.

6.Post-Processing (Non-Maximum Suppression)

To remove duplicate or overlapping detections, Non-Maximum Suppression (NMS) is applied. This step selects the best bounding box based on confidence scores and eliminates redundant detections, ensuring clean and accurate results.

7.Classification and Labeling

Detected objects are classified into predefined categories such as Grocery, Medical, and Household. Each detected object is assigned a label along with a confidence score indicating the accuracy of prediction.

8.Output Generation

The final output is displayed with bounding boxes, class labels, and object counts. The system presents results in real time, enabling users to monitor detected products efficiently. The output can also be stored for further analysis or inventory management.

Software implementation

The software implementation of the proposed YOLOv8-based Object Detection System integrates Python programming, deep learning frameworks, and computer vision libraries into a cohesive real-time application. The system is implemented using Python and utilizes libraries such as YOLOv8, OpenCV, NumPy, and Matplotlib to perform object detection, classification, and visualization efficiently. The system is designed to process images and live camera feeds to detect and categorize objects in real time.

Data Collection and Preprocessing

The first step in software implementation involves collecting a dataset of grocery and related objects using cameras or publicly available datasets. The captured images or video frames are preprocessed to improve quality and consistency. Techniques such as resizing, normalization, noise reduction, and image enhancement are applied.

Data augmentation methods such as rotation, flipping, and scaling are used to increase dataset diversity. This helps the model generalize better and improves accuracy. Proper labeling of objects into categories such as Grocery, Medical, and Household is essential for supervised learning. The processed data is then divided into training and testing sets. Efficient preprocessing ensures reliable input for the detection model and plays a crucial role in overall system performance.

Model Development using Deep Learning

In this phase, a deep learning model based on YOLOv8 is designed and implemented. YOLOv8 is a single-stage object detection algorithm that performs detection and classification in a single pass, making it highly efficient for real-time applications.

The model architecture consists of multiple components such as convolutional layers for feature extraction, feature aggregation layers, and detection layers for predicting bounding boxes and class labels. Frameworks and libraries in Python are used to train the model on labeled datasets. The model learns object features through iterative training. Hyperparameters such as learning rate, batch size, and number of epochs are tuned to achieve better performance. This phase is critical for accurate object detection. A well-trained model ensures high detection accuracy and faster processing.

Real-Time Object Detection

The system uses computer vision techniques to detect objects in real time using a webcam or live video feed. OpenCV is used for capturing video frames and processing them continuously.

Each frame is passed to the YOLOv8 model, which detects objects and generates bounding boxes, class labels, and confidence scores. Real-time detection requires efficient processing to minimize delay. Frame-by-frame analysis ensures smooth and continuous detection.

This module acts as the interface between the real-world environment and the detection system. It enables practical implementation in scenarios such as retail stores and warehouses.

Object Classification and Counting

Once objects are detected, they are classified into predefined categories such as Grocery, Medical, and Household. The model predicts the most probable class label based on learned features. Each detected object is assigned a confidence score indicating the accuracy of prediction. The system also counts the number of detected objects, which is useful for inventory management and automation. The classification process

ensures that detected objects are meaningful and usable for real-world applications. Accuracy at this stage is essential for reliable system performance.

Output Generation

The final step is to generate user-friendly output based on detected objects. The system displays bounding boxes, labels, and object counts on the screen. Visualization tools such as OpenCV and Matplotlib are used to present results clearly. The system may also store detection results in an SQLite database for future analysis or record keeping inclusive and accessible. The speech output allows communication with people who do not understand sign language. The system ensures minimal delay between gesture recognition and output generation. A simple user interface can be designed for better interaction. This module enhances usability and real-world applicability. It completes the communication cycle effectively. The combination of text and speech improves overall user experience.

System Workflow

The complete software workflow of the implemented system is as follows:

- The system initializes the camera or input source to capture images or live video
- Continuous frames are captured for real-time processing
- Each frame is passed to the preprocessing stage for resizing and normalization
- The processed image is fed into the YOLOv8 model
- The model performs feature extraction and detects objects in the image
- Bounding boxes, class labels, and confidence scores are generated
- Non-Maximum Suppression is applied to remove duplicate detections
- Detected objects are classified into predefined categories
- The system counts the number of detected objects
- The final output is displayed to the user in real time with visualization

Results and Discussion

The proposed YOLOv8-based Object Detection System was implemented and tested in a real-time environment using images and live camera feeds containing multiple objects from categories such as Grocery, Medical, and Household. The system was evaluated based on parameters including detection accuracy, response time, real-time performance, robustness under different conditions, and overall system stability.

Experimental Setup

The experimental setup of the proposed object detection system consists of both hardware and software components required for real-time detection and classification. A webcam or camera is used to capture live images and video streams. The system is implemented on a computer with moderate processing power, such as an Intel i5 processor with 8 GB RAM. The software environment includes Python programming language along with libraries such as YOLOv8, OpenCV, NumPy, and Matplotlib. A labeled dataset of grocery and related objects is used to train and test the model. The system is designed to handle real-time input and provide accurate detection results.

Real-Time Performance

The system is designed to operate in real time, ensuring quick detection and classification of objects from live video feeds. The use of the YOLOv8 algorithm enables fast processing, as it performs object detection in a single pass. During testing, the system was able to detect multiple objects simultaneously with minimal delay. The real-time performance was smooth and efficient, making it suitable for applications such as retail stores and inventory management. The optimized model reduces computational overhead and improves speed. However, performance may vary depending on hardware specifications such as processor speed and availability of GPU. Even with moderate hardware, the system provides satisfactory real-time detection, making it accessible and practical.

Response Time Analysis

The response time of the system is defined as the time taken to process an input image or frame and generate detection results. Experimental observations show that the system achieves response times in the range of a few milliseconds to around 1–2 seconds, depending on system configuration and input complexity. The use of YOLOv8 significantly reduces latency due to its efficient architecture. Faster hardware and GPU support further improve response time. Slight delays may occur when multiple objects are present or when the scene is complex. Overall, the system maintains a good balance between speed and accuracy, ensuring smooth real-time performance.

Robustness in Different Conditions

The system was tested under various environmental conditions such as different lighting levels, background complexity, and object arrangements. The results show that the system performs well in controlled environments with proper lighting and clear object visibility. However, performance may decrease slightly in low-light conditions, cluttered backgrounds, or when objects are partially occluded. Variations in object size and orientation can also affect detection accuracy. To improve robustness, preprocessing techniques such as image normalization and resizing are applied. Training with diverse datasets and data augmentation helps the model generalize better. Despite some limitations, the system performs reliably in most real-world scenarios.

Discussion

The proposed YOLOv8-based Object Detection System demonstrates effective performance in detecting, classifying, and counting objects in real time. The integration of deep learning and computer vision techniques improves both accuracy and speed. The system performs well under normal conditions and provides fast and reliable results. It significantly reduces manual effort in tasks such as product identification and inventory management. However, certain limitations are observed in challenging conditions such as poor lighting and cluttered environments. The system is user-friendly and can be easily deployed in practical applications such as retail stores and warehouses. Future improvements may include enhancing model accuracy, improving performance in complex environments, and optimizing the system for low-end devices.

Conclusion

The development of the AI-Based Object Detection, Counting, and Sorting System represents a significant advancement in automating inventory management and object recognition processes. This research focused on designing a system that leverages computer vision and deep learning techniques, specifically the YOLOv8 algorithm, to detect, classify, and count objects in real time. The proposed system effectively reduces manual effort and improves efficiency in environments such as retail stores, warehouses, and industrial setups.

By utilizing advanced image processing techniques and deep learning models, the system demonstrates the ability to accurately identify multiple objects simultaneously and classify them into predefined categories such as Grocery, Medical, and Household. The integration of real-time detection ensures faster processing and immediate output generation, making the system practical for real-world applications. Additionally, the object counting feature enhances its usability in inventory tracking and management tasks. The proposed system is cost-effective, user-friendly, and adaptable to various environments. It minimizes human errors, improves operational efficiency, and supports automation in inventory-related processes. The system can be easily integrated into existing workflows, making it a valuable solution for businesses aiming to adopt smart technologies. However, certain challenges such as varying lighting conditions, cluttered backgrounds, and occlusion of objects may affect detection accuracy. These limitations highlight the need for further improvements and optimization. Despite these challenges, the system performs reliably under most practical conditions. In the future, the system can be enhanced by expanding the dataset to improve accuracy in complex environments and by adding more product categories for better scalability. Integration of barcode and QR code recognition can further improve product identification and reduce misclassification. The system can also be extended with real-time object tracking to monitor moving items in warehouses or production lines. Overall, this research contributes to the advancement of intelligent automation systems and provides a foundation for developing smart inventory management solutions. The proposed system has strong potential to evolve into a fully automated and intelligent platform, supporting the vision of efficient, scalable, and technology-driven industrial environments

Acknowledgment

The authors would like to express their sincere gratitude to Prof. Neelam Jadhav, project guide, for her invaluable guidance, continuous support, and constructive feedback throughout the development and implementation of this AI-based object detection, counting, and sorting system. Her insights and encouragement greatly contributed to the successful completion of this project. We also extend our sincere thanks to the Head of the Department, Prof. Nilima Patil, and the Department of Computer Engineering, Genba Sopanrao Moze College of Engineering, Balewadi, Pune, for providing the necessary resources, infrastructure, and a supportive academic environment to carry out this project. We acknowledge Savitribai Phule Pune University for encouraging research-oriented learning and providing access to the required tools and facilities. Finally, we thank our peers and team members for their cooperation and contributions during the development of this project.

References

1. D. Liang, Q. Geng, Z. Wei, D. A. Vorontsov, E. L. Kim, M. Wei, and H. Zhou, "Anchor Retouching via Model Interaction for Robust Object Detection in Aerial Images," IEEE Transactions on Geoscience and Remote Sensing, vol. 60, 2022, Art. no. 5610013.
2. P. Zhao, Z. Qu, Y. Bu, W. Tan, and Q. Guan, "PolarDet: A Fast, More Precise Detector for Rotated Target in Aerial Images," International Journal of Remote Sensing, vol. 42, no. 15, pp. 5831–5861, Aug. 2021.
3. X. Yang, X. Yang, J. Yang, Q. Ming, W. Wang, Q. Tian, and J. Yan, "Learning High-Precision Bounding Box for Rotated Object Detection via Kullback–Leibler Divergence," in Proceedings of Advances in Neural Information Processing Systems, vol. 34, 2021, pp. 18381–18394.
4. X. Yang, J. Yan, Q. Ming, W. Wang, X. Zhang, and Q. Tian, "Rethinking Rotated Object Detection with Gaussian Wasserstein Distance Loss," in Proceedings of the International Conference on Machine Learning, 2021, pp. 11830–11841.
5. X. Yang, Y. Zhou, G. Zhang, J. Yang, W. Wang, J. Yan, X. Zhang, and Q. Tian, "The KFIoU Loss for Rotated Object Detection," arXiv preprint arXiv:2201.12558, 2022.
6. J. Han, J. Ding, N. Xue, and G.-S. Xia, "ReDet: A Rotation-Equivariant Detector for Aerial Object Detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2021, pp. 2785–2794.
7. M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10781–10790, 2020.
8. H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2881–2890, 2017.
9. Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.
10. D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, 2004.