Crowd Counting in Restaurants Using Computer Vision

Aaditya Gole Daksh Rajesh

Project Code

Contents

Abstract

Crowd counting is a significant problem in computer vision, particularly in dynamic environments like restaurants where both customers and employees are present. This report presents a structured approach to solving the problem of accurately counting people (customers and employees) in a restaurant setting using video footage. The task is performed using machine learning models and computer vision techniques, iteratively improving the results to overcome challenges such as bounding box flickering (leading to an unstable count), over-detection, and distinguishing between customers and employees. Two main approaches were tested: the YOLOv8 model and a Faster R-CNN model with Non-Maximum Suppression (NMS). The report explains the methodologies, challenges faced, and how the issues were progressively addressed.

1. Problem Statement

The primary objective of this project is to accurately count the number of people present in a cafe in real-time, while excluding the employees from the count. This involves distinguishing between customers and employees in a dynamic environment using video footage. The challenge is to ensure that the count is stable and accurate, despite the presence of overlapping bounding boxes, flickering counts, and the need to differentiate between customers and employees.

2. Introduction

Crowd counting is a critical task with applications in domains such as retail management, safety monitoring, and resource optimization. The goal of this project is to count the number of people accurately in a restaurant video, distinguishing between customers and employees, and minimizing flickering or over-counting caused by overlapping or multiple bounding boxes.

The primary challenges include:

- **Overlapping bounding boxes:** causing inaccurate counts.
- **Flickering counts:** due to frame-to-frame inconsistencies.
- **Distinguishing customers from employees:** in the video provided.

We address these challenges through iterative refinement using machine learning models and techniques such as supervised polygon masking and suppression mechanisms.

3. Methodology

This problem was solved in three iterations. In each iteration, we addressed the challenges faced in the previous step.

3.1. YOLOv8 Pretrained Model

3.1.1. YOLO Architecture

YOLO (You Only Look Once) is a real-time object detection model known for its speed and simplicity. Unlike region-based methods, YOLO treats object detection as a single regression problem. It divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell simultaneously.

Key components of YOLO architecture include:

- **Convolutional Backbone:** Extracts features from the input image.
- **Unified Detection:** All predictions (class, location, and confidence) are generated in one pass through the network.

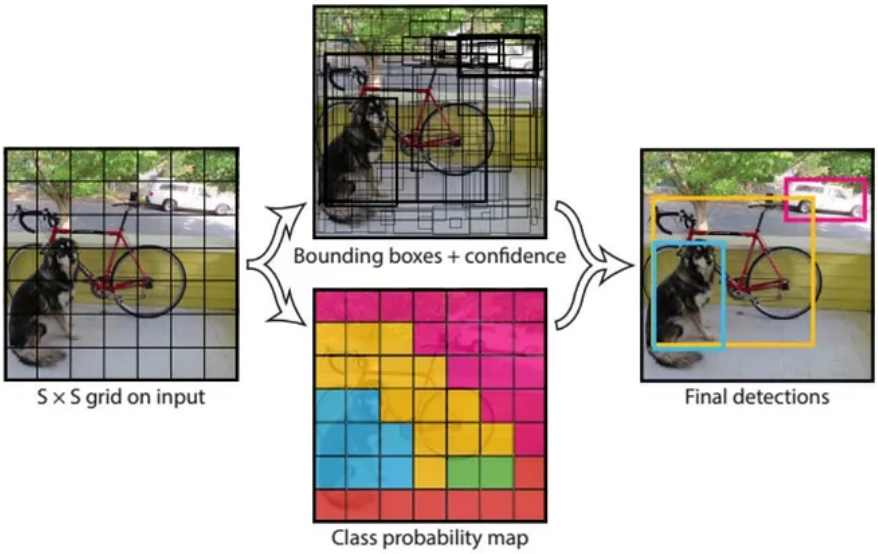

Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

Figure 1: How yolo detects images).

Advantages:

- High inference speed, suitable for real-time applications.
- End-to-end training, reducing complexity.

Challenges:

- Struggles with detecting small objects in crowded scenes.
- Generates multiple overlapping bounding boxes, requiring post-processing such as Non-Maximum Suppression (NMS) for refinement.

3.1.2. Implementation and Results

The YOLOv8 pretrained model was selected for object detection due to its speed and efficiency. The methodology and outcomes are as follows:

- **Detection and Counting:** While the YOLOv8 model was able to successfully detect a person when present, it also provided multiple bounding boxes for each person, which caused over-counting. The count flickered between frames due to inconsistencies in bounding box generation.

Challenges Faced:

- Over-detection: Multiple bounding boxes on a single individual.
- Flickering Counts: Frame-to-frame inconsistencies in detection.
- Misclassification: Employees detected within the polygon, leading to customer miscounts.

3.2. Polygon-Based Zone Restriction

3.2.1. Implementation and Results

To minimize detections of employees and focus on customers, a polygonal area was defined to specify the "customer area" within the video frame. This zone was expected to contain only customers, as employees typically remained outside this area. The implementation involved using supervision to create the polygonal zone and restrict detections within it.

However, this approach had limitations. Employees, such as cleaning staff, would occasionally enter the customer area, leading to false-positive detections. This made it challenging to accurately distinguish between customers and employees based solely on their location in the frame.

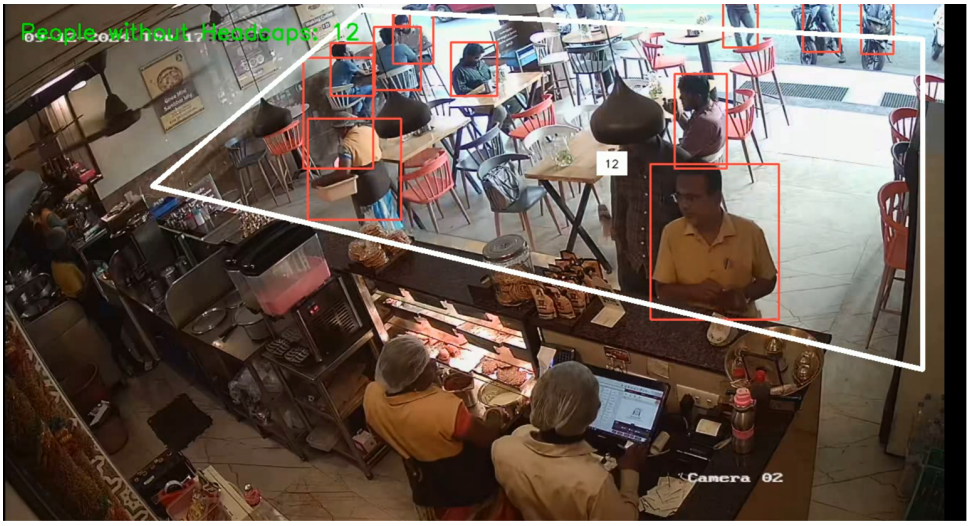

Figure 2: Detection using polygon-based zone restriction (placeholder image).

3.3. Employee Headcap Detection and Subtraction

3.3.1. Implementation and Results

To effectively distinguish between customers and employees, an additional model was employed to detect employees based on their distinctive headcaps, which all employees wear in the video. The approach involved detecting all people in the frame and then subtracting those identified as employees.

Furthermore, to address the flickering problem caused by rapid detection and loss of detection, the average count over the last 10 frames was calculated to stabilize the count.

The implementation can be found in the python notebook in the github link

This approach effectively differentiated customers from employees, resulting in a more accurate customer count. Averaging over frames also mitigated the flickering issue, providing a stable count over time.

3.4. Challenges Addressed

- **Distinguishing between employees and customers:** By detecting employees based on their headcaps and subtracting them from the total people detected.
- **Flickering Counts:** Stabilized the count by averaging over the last 10 frames.

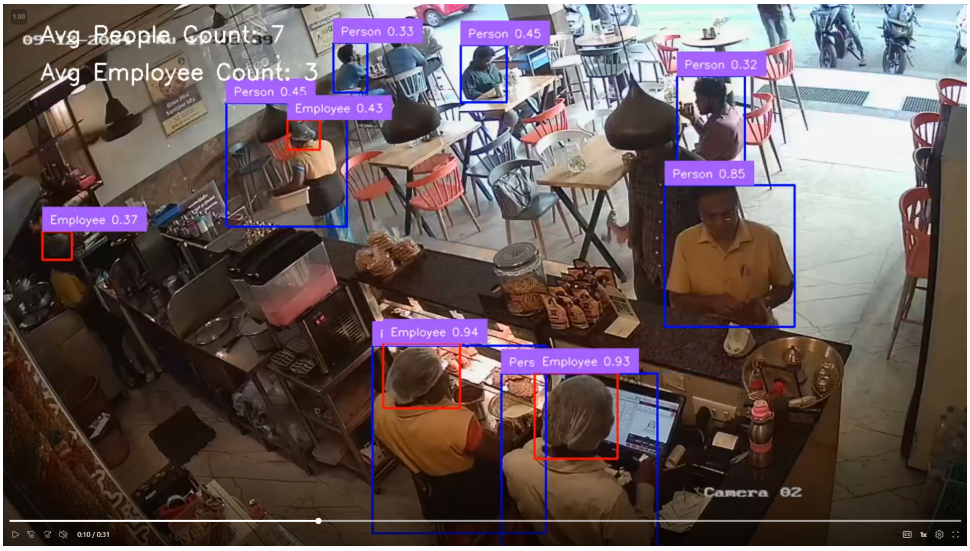

Figure 3: Final detection with employee subtraction and stabilized counts (placeholder image).

4. Conclusion

This project highlights the iterative refinement required to tackle real-world challenges in crowd counting. By implementing polygon-based zone restriction and employee headcap detection, we significantly improved detection stability and accuracy. Future work will focus on enhancing model generalization and addressing occlusions to further improve the system's robustness.

5. References

- Lin, T.-Y., et al. "Faster R-CNN with Feature Pyramid Networks." *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- Ani Agarwal, ***YOLO explained***