Empty Seat Detection in Campus Restaurants for IISER - B Community Using Computer Vision

Written by Siddhi Pravin Lipare and Aryan Jain
3rd year, Data Science and Engineering Major, IISER Bhopal

Intermediate Project Report

Problem Statement

- Our objective is to develop a Computer Vision-based System that captures live footage from a camera within Campus Restaurants and detects unoccupied seats.
- **Step 1**: The first stage involves assessing the presence of a person within a defined bounding box.
- **Step 2**: Subsequently, we proceed to verify the presence of a chair within a designated bounding box.
- **Step 3**: In the final step, if the bounding boxes corresponding to a person and a chair intersect, the seat is considered occupied (indicated in red); otherwise, it remains unoccupied (indicated in green).

Some notable findings

- **Initial Research**: We conducted an in-depth analysis of Chia-Hung Lin's work on "Library empty seat prediction"[2]. He used Deep Convolutional Neural Networks, with a specific focus on Mask R-CNN, to classify bounding boxes. He implemented TensorFlow's object detection API, which is a great choice for working with pre-trained models. However, it's important to note that this API may not be the most suitable option for real-time object detection applications due to heavy computational load and lack of optimisation.

- We also read an informative IEEE paper [4] which has a detailed analysis of the process carried out during Object Detection (Figure 1) using YOLO. They have used both YOLOv7 and YOLOv8 for testing the differences in their accuracy, precision, recall and F1 score. Even though YOLOv7 and YOLOv8 produce a high accuracy, they tend to occasionally produce false positives and false negatives. Furthermore, a large amount of annotated data is required in order to detect the Objects accurately.

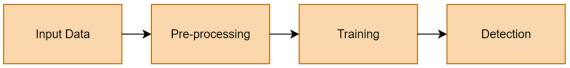

Figure 1: General Process of Object Detection

- **Algorithm Finalisation**: We decided to search more about YOLO (You Look Only Once), a pre-trained object detection model, which works great for the COCO Dataset which has 80 labels, which includes chair, human, dining table, water bottle and even a cell phone (in case someone has kept a cell phone/bottle on the table to reserve his/her space.
- **Principle**: YOLO basically applies a single neural network to the entire image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. We also tried implementing this model on a short

Figure 2: Using YOLO to detect person and chair in Sudarshan Restaurant

video (Figure 2).

- **Labelling**: As we tried implementing YOLO for object detection, we observed that it displayed bounding boxes around every detected person. However, this approach led to a cluttered visual display. Furthermore, our primary objective is to identify empty chairs, making the display of bounding boxes around individuals unnecessary. To enhance clarity and simplify the recognition of seat occupancy, we have chosen to display bounding boxes exclusively around chairs. This will allow us to easily distinguish between occupied and unoccupied seats. To implement this, we need to selectively download specific labels while discarding others. This task can be efficiently accomplished using the FiftyOne package.

- **Model Version**: We conducted thorough research across various websites and research papers to determine which version of YOLO to use. Although YOLOv8 is the most recent version of YOLO, launched on January 10th, 2023, we were skeptical about using the latest version due to its reported slower detection speed and the challenges it presents in terms of optimization. Consequently, we have decided to utilize YOLOv7, which is currently recognized as the fastest and most powerful object detection algorithm.

- **Alternative approach**: Another method for addressing this project involves training the YOLO Algorithm on a custom dataset. This can be done by implementing the following steps:

  1. **Data Acquisition**: Acquiring a collection of images corresponding to two labels: ”Empty” and ”Occupied”.
  2. **Image Annotation**: Each acquired image must be precisely annotated. The process involves correctly marking objects of interest, defining their spatial coordinates, and providing accurate classifications (Figure 3).

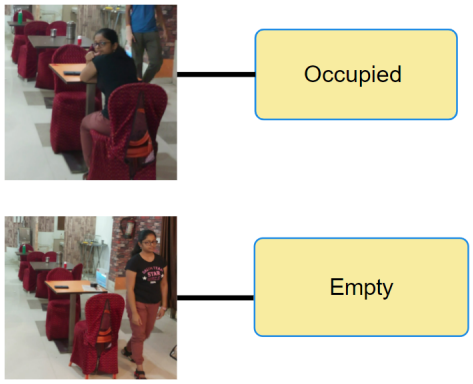

Figure 3: Labels

  3. **Model Training**: With an annotated dataset in hand, we can move on to the training stage. Leveraging the YOLO algorithm, we’ll train the model to recognize and accurately identify objects within our custom dataset.

- **Evaluation Metrics**: Model evaluation can be done using a confusion matrix to measure the number of correct and incorrect predictions. There are 4 types of evaluation matrix that will be tested on the model:

  1. **Accuracy**: Measures the extent to which the detection model successfully predicts correctly overall.
  2. **Precision**: Measures the extent to which objects

detected by the detection model are correct. Precision provides an indication of how well the model can avoid detecting fake objects.

3. **Recall**: Gives an indication of how well the model can capture all the objects that are actually in the image.
4. **F1 Score**: The harmonic mean value of precision and recall is used when we want to achieve a balance between the two metrics

Following this method, we can make the YOLO Algorithm work for our specific needs. However, if we use a custom dataset and try to expand our project for other places in campus such as ICH or Library for the purpose of empty seat detection, then we'll have to repeat the process for creating a custom dataset. This might become a tedious process but will generate a higher accuracy if time permits.

Individual Roles

Member 1: Siddhi Lipare

- Analysed various YOLO versions to choose the most optimised one, and also implemented it on local machine for a video taken by us in Sudarshan Restaurant.
- Implementing YOLO algorithm for particular labels and generating a bounding box.
- Creating a custom database and manually annotating them. (Unoccupied table, singly occupied, doubly occupied and so on).

Member 2: Aryan Jain

- Found an alternative approach for our problem statement (How to implement the YOLO Algorithm in an optimised manner using a custom dataset).
- Implementing YOLO algorithm for particular labels and generating a bounding box.
- Searched for various tools that can download the data from particular labels of different datasets.

References

[1] Downloading a specific part of a dataset: Stack Overflow

[2] Seat Status Detection in Library by Chia-Hung Lin: GitHub

[3] Intelligent System for Monitoring the Availability of Bus Passenger Seats using the YOLO Method : IEEE

[4] Deep Learning-Based Object Detection Algorithms on Image and Video: IEEE

[5] YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors : YOLOv7

[6] YOLOv7: The Most Powerful Object Detection Algorithm (2023 Guide) YOLOv7

[7] UAV-YOLOv8: A Small-Object-Detection Model Based on Improved YOLOv8 for UAV Aerial Photography Scenarios: YOLOv8