CENG313 DATA SCIENCE

EMPTY PARKING SPACE FINDER

PROJECT REPORT

Tolga SEYMEN

Muhammed AKSOY

Yakup KARATAŞ

Instructor: Prof. Dr. Şeref SAĞIROĞLU

1. Introduction

This project aims to find a creative solution which contains data science applications for the empty parking space problem. Our motivation to choose this problem is combining image processing techniques and programming microcontrollers which is a new area to learn for us. Our project scope is to design a cost-effective, real-time parking monitoring system by integrating ESP32-CAM for image acquisition with a custom-trained YOLOv11 model for precise occupancy detection.

2. Dataset

For our dataset, we used a 681 MB collection from Roboflow which contains 7017 trains, 558 valid and 226 test images. The dataset features parking spots captured from various angles and distances (near vs. far). All the images in the dataset are sized 640x640 which is appropriate for the YOLOv11 model's requirements.

3. Data Preprocessing

Luckily, we didn't perform a lot of data preprocessing because our dataset is almost %97 clean. Luckily, there are not a lot of raw or corrupted images in our dataset. We found the raw or corrupted images, and deleted them. The data preprocessing we used in our project is deleting raw or corrupted images.

4. Methods Used

We used the YOLOv11 model to develop our project. The YOLOv11 model is using CNN (Convolutional Neural Network) architecture. This is an architecture which is learning from a dataset directly. It's usually useful for classification, detection or categorization from images, such as detecting cars in an image. We optimized our model using fine-tuning, a technique where we adapted a pre-trained AI model to our specific needs by retraining it on our custom parking dataset, rather than starting from zero.

5. Results

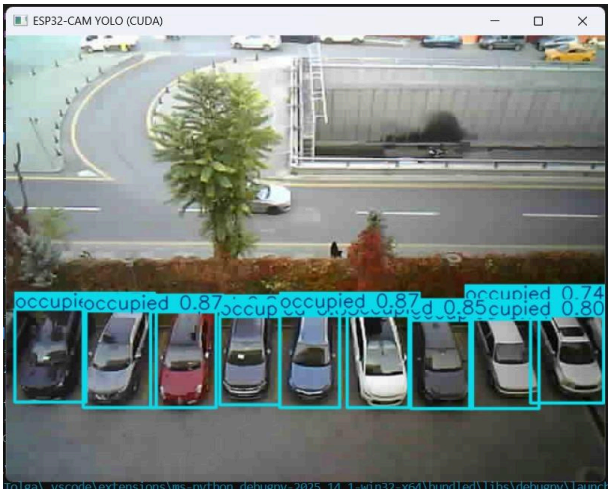

Figure 1

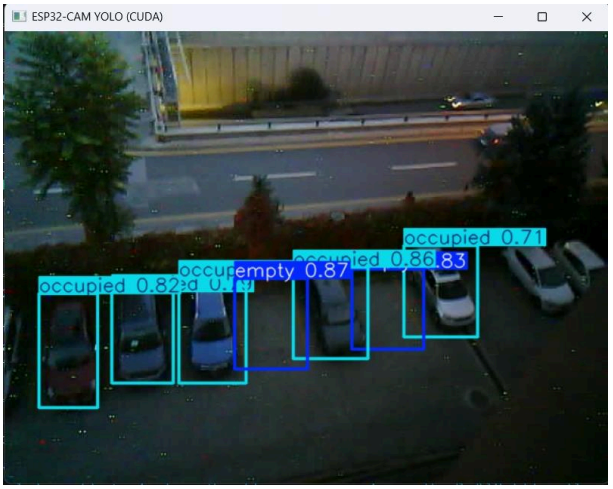

Figure 2

```
WARNING ⚠ Box and segment counts should be equal, but got len(segments) = 8307, len(boxes) = 11950. To resolve this only boxes
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 35/35 5.7it/s 6.2s
                   all        558      11950      0.961      0.947      0.978      0.894
                 empty        503       3650      0.946      0.932      0.971      0.852
              occupied        484       8300      0.977      0.962      0.986      0.936
Speed: 0.9ms preprocess, 1.2ms inference, 0.0ms loss, 4.7ms postprocess per image
Results saved to /content/runs/detect/val
mAP50-95: 0.8940554413929055
```

Figure 3

As you can see, in Figure 1 and Figure 2, our model is working with at least %80 accuracy rate. In Figure 3, you can see the values of mAP50 and mAP50-95 which show the reliability of our model are appropriate to use for academic research.

6. Evaluation

According to the results, our model reached at least %80 accuracy rate which is a good rate for the beginning. However, when we tried to get a live stream from our camera (ESP32-CAM OV2640) the live stream's FPS was sometimes reduced immediately or the program crashed. Thanks to the quality settings of the camera, we reduced our camera's quality, and got better FPS.

This project's strengths are mobility, accuracy and real-time accessing. We can connect the camera to a powerbank, and move the camera module approximately 3-5 meters. The model's accuracy rate is at least %80 which is a good rate for a project. If you start a live stream from the camera, you can use the model to process the live stream which is good for security cameras.

Finally, this project's weakness is dropping FPS suddenly, and crushing sometimes.