Please answer these questions in your lab journal. Include screenshots of your results where appropriate.

Part 1: Conceptual Understanding

1. **Explain in your own words:** What is the difference between image classification and object detection?

The difference between image classification and object detection is that image classification identifies what is in the image, while object detection is also focused on where a piece of an image is along with what is it.

2. **Bounding boxes:** Why do we use fractions (0 to 1) for bounding box coordinates instead of pixel values?

Fractions make it easier to set up the dimensions of the bounding box in relation to the image. It also makes it easier to read and change the bounding box coordinates, and it makes scaling the bounding box in relation to the image easy.

3. **IoU metric:** Explain what IoU measures. Why is an IoU of 0.5 commonly used as a threshold?

An IoU of 0.5 offers a decent overlap of 50% and anything better than that is deemed satisfactory.

Part 2: Implementation Experience

4. **Visualization function:** What was the most challenging part of implementing the `display_examples` or `plot_detections` function? How did you solve it?

I didn't suffer any challenges when implementing the `display_examples` and `plot_detections`.

5. **IoU calculation:** Walk through your IoU implementation. What would happen if you forgot to use `max(0, ...)` for the intersection width and height?

For the IoU implementation I had to find the intersection rectangle by finding a new bounding box with each side of the box being the one out of the two from both bounding boxes that is closest to the center. Then using this new bounding box, I have to calculate the intersection area by finding the width and height of the bounding box. `max(0, ...)` should also be utilized

when finding the width and height of the bounding box otherwise if the boxes don't overlap, the width or height of the intersection area will be negative. Then we calculate the union area by adding the areas of the two previous boxes together. Finally, we calculate the IoU by dividing the intersection area with the union area.

6. **Debugging:** Did you encounter any errors while coding? What was the error and how did you fix it?

Not really.

Part 3: Results Analysis

7. **Confidence threshold experiment:**
   - How many boxes did you see with threshold=0.3 vs threshold=0.7?
   - Which threshold do you think is better? Why?

   When the threshold was 0.3, I saw 12 boxes in total. When the threshold was 0.7, there were two. Personally, I'd go with the 0.3 threshold, or at least something of that range, because it gives the model more breathing room, although it might come at a cost as aspects of the image can be more misinterpreted.

8. **IoU threshold experiment:**
   - How did precision and recall change when you changed the IoU threshold?
   - Explain why this happened.

   When the threshold was low, there were more bounding boxes, and thus more objects can be identified. As such recall increased. However, this also weakened the confidence results as now there are more opportunities to get it wrong, decreasing the precision as a result.

   When the threshold was high however, there are fewer bounding boxes, and as such fewer objects to identify, decreasing the recall. However, this leaves guesses that bring higher confidence, increasing precision as a result.

9. **Model errors:** Find one example each of:
   - A **false positive** (model detected something incorrectly)
   - A **false negative** (model missed an object)
   - Include screenshots and explain why you think the model made these errors.

When the threshold was 0.3, there were a lot of false positives in its guessing because there were a lot of bounding boxes in areas that don't necessarily add to the whole image.

Meanwhile, when the threshold was 0.7, there were a lot of false negatives as there aren’t enough bounding boxes to adequately designate important parts of the image to their proper descriptions.

Part 4: Critical Thinking

10. **Precision vs Recall trade-off:**

    - For a self-driving car detecting pedestrians, would you prioritize high precision or high recall? Why?

I would prioritize high recall since it would be crucial to find every single pedestrian on the road. And even if the prediction would be incorrect, there is a good chance that it's going to be something else to avoid.

- For a photo app that tags objects in your pictures, would you prioritize high precision or high recall? Why?

I would probably prioritize high precision because I would want to give any object that has been identified, a consistent label. I'm not sure if finding all the items would be necessary here, especially if that would only cover the photo even more.

11. **Pre-trained models:**

    - What are the advantages of using a pre-trained model?

Using a pre-trained model saves time and money and can achieve higher accuracy due to it already being trained on large datasets.

- What are the disadvantages?

However, it can be limited when it comes to flexibility, has the potential for inherent biases, and can have issues with domain mismatch or general lack of customization for highly specific tasks.

12. **Real-world applications:** Name three real-world applications of object detection and explain why object detection (not just classification) is necessary for each.

Object detection has been used in:

- **Navigation: Identifying and tracking objects in their environment, such as pedestrians, other vehicles, and traffic signs**
- **Medical Diagnostics: Identifying anomalies such as tumors, lesions, or fractures in medical scans like X-rays, MRIs, and CT scans.**
- **Inventory Management: Identifying misplaced or out-of-stock items, aiding in managing inventory more efficiently.**

Part 5: Going Further (Optional - Bonus Points)

13. **F1 Score:** Research the F1 score metric. Implement a function to calculate it and add it to the evaluation. What does the F1 score tell you that precision and recall alone don't?
14. **Other models:** Research another object detection model (e.g., YOLO, Faster R-CNN). How does it differ from SSD MobileNet V2?
15. **Your own images:** Try running the detector on your own images (if possible). How well does it perform? What types of objects does it detect well vs poorly?