Erwin Cheng

ITAI 1378

October 16, 2025

Journal / Summary

Patricia Mcmanus

Lab 06 Journal: Object Detection with Transfer Learning

Part 1: Conceptual Understanding

The way I understand it, image classification gives you one label for the whole image, It only tells you what object is in the image (e.g., "This is a picture of a dog"). Object detection is much more detailed. It scans the image and draws boxes around all the different things it recognizes, giving each box a label, So instead of one label per image, you get multiple labeled regions. (e.g., "This is a 'dog' in this box, and that's a 'car' in that box over there"). So, it’s the difference between a summary and a detailed report.

Bounding box coordinates are usually given as fractions (from 0 to 1) relative to the image width and height. This makes the coordinates scale-independent — the model can handle any image size or resolution without recalculating pixel-specific values. If we used pixels, we'd have to recalculate the box for every single image size, which would be a huge pain. This method makes the model's output universal.

IoU (Intersection over Union) measures how well the predicted box overlaps with the actual (ground truth) box. It’s calculated as the area of overlap divided by the area of union between the two boxes.

An IoU of 0.5 is often used as a threshold because it’s a balanced middle ground — anything lower means the detection is too far off, while higher thresholds may be too strict and penalize slightly off but still useful predictions.

Part 2: Implementation Experience

Visualization Function

The most challenging part was getting the bounding boxes to line up correctly on the images when using the display_examples or plot_detections function. The boxes initially appeared shifted or stretched because the coordinates needed to be multiplied by the image’s width and height. Once I realized the model used normalized coordinates, I adjusted them before drawing the rectangles, which fixed the issue.

IoU Calculation

My IoU function calculated the intersection by comparing the edges of two boxes. Without max(0, …) for the intersection width and height, the code could produce negative values when boxes didn't overlap — which would cause negative IoU scores. Adding max(0, …) ensures that non-overlapping boxes return an intersection of zero.

Debugging

At one point, I got a ValueError when trying to unpack detection results from the TensorFlow model — it turned out the model's output was a dictionary, not a list. Once I accessed the correct keys (like 'detection_boxes' and 'detection_scores'), the visualization worked as expected.

Part 3: Results Analysis

Confidence Threshold Experiment

With a threshold of 0.3, I saw many bounding boxes — including several false positives. At 0.7, there were fewer boxes, but the detections were more accurate and relevant. Overall, 0.7 felt like the better balance because it reduced noise while keeping confident predictions.

IoU Threshold Experiment

When I increased the IoU threshold, precision went up (fewer false positives were counted as correct), but recall dropped (more true objects were missed). This happened because the higher IoU standard made the system stricter about what counts as a "correct" detection.

Model Errors

- False Positive: The model once detected a "person" in a tree shadow — probably because the shape resembled a human outline.
- False Negative: In one image, a small dog wasn't detected because it was partly obscured and too far away; the model likely missed it due to its low confidence score.

Part 4: Critical Thinking

Precision vs Recall Trade-off

- For a self-driving car, I'd prioritize high recall — missing a pedestrian (false negative) is far worse than detecting an extra one (false positive).
- For a photo tagging app, I'd prioritize high precision — users prefer accurate, clean results rather than messy tags that misidentify objects.

Pre-trained Models

- Advantages: They save tons of time and computing power since they've already learned rich visual features from large datasets like COCO. You can fine-tune them for your task instead of starting from scratch.
- Disadvantages: They may not generalize perfectly to your specific dataset or image style. You're also limited by the classes they were originally trained on.

Real-world Applications of Object Detection

- Autonomous Vehicles: Detecting pedestrians, traffic signs, and other cars — classification alone isn't enough because exact positions are crucial for safety.
- Security Cameras: Identifying and tracking people or objects in live video feeds requires knowing where movement occurs, not just what it is.
- Retail Analytics: Detecting how products are placed or when shelves are empty relies on locating each item precisely within the frame.

Part 5: Going Further

F1 Score

The F1 score combines precision and recall into a single metric, using the harmonic mean. It's especially useful when one metric alone doesn't tell the full story — for example, a model with high precision but low recall isn't actually performing well overall.

Other Models – YOLO vs SSD MobileNet V2

YOLO (You Only Look Once) processes the entire image in one pass, making it extremely fast and efficient for real-time detection. SSD MobileNet is lighter and designed for mobile devices but may be slightly less accurate or slower compared to newer YOLO versions.

Testing on My Own Images

When I tried it on my own images, it performed quite well on common objects like people and cars but struggled with less common ones or images with poor lighting. Overall, the pre-trained model did a solid job for general-purpose detection.