**Reflective Journal**

In this lab, we switched from just classifying images to actually detecting and locating objects using bounding boxes. The goal was to understand how object detection works, try it out with a pre trained model and break down what’s happening under the hood

**A screenshot of a computer program

AI-generated content may be incorrect.**

The beginning of the process which was really just downloading and prepping the images for use

**A screenshot of a computer program

AI-generated content may be incorrect.**

This was my first time seeing the Hub.load function and I didn’t know so seeing the breakdown helped

A screenshot of a computer

AI-generated content may be incorrect.

Heres an example of a predicted and ground truth rendered or processed through the model.

I did however have trouble running my own image because the heatmap AND the run\_detector function were missing from my set I was unsure if it was intentional or not and I was supposed to code that part if I was please let me know and I will redo that part of the assignment

**What's the main difference between image classification and object detection?**  
Classification tells you what's in the image. Detection tells you what and where. We saw boxes and labels show up, not just names.

**Why'd we pick SSD MobileNet V2?**  
It's fast and lightweight. Works well on low power. Not the most accurate but it’s good for quick stuff.

**What does find\_images\_with\_classes do?**  
This function filters images to include only those that contain the target classes. It is especially useful with large datasets like COCO or Pascal VOC, where we may only want to focus on a subset

**How does threshold=0.5 affect detections?**  
It hides the things the model’s accuracy isn’t too confident or positive about. Higher threshold = fewer boxes. Lower = more boxes, even wrong ones.

**What’s the heatmap for?**  
Shows where the model's looking. Brighter spots would mean it's more confident there is something.

**What objects got detected best? Which didn’t?**  
People and cars were solid. Small stuff or cluttered scenes messed it up.

**Any wrong boxes or misses? Why?**  
Yes some boxes were off or missed the object. Could be lighting, object was tiny, or too much going on.

**Would full Pascal VOC help?**  
It might because more data means better learning. It’d probably miss less.

**How would you make it detect just animals or cars?**  
Change the class list to only include those, adjusting the filtering logic in (find\_images\_with\_classes), we can tailor detections to specific domains. Filter the rest out in the code.

**How would you train your own detection model?**  
Get images then label them, pick a model and finally train it. It would take a lot more date

**Where’s SSD MobileNet V2 useful?**  
It would be useful in a broad sense. Phones, security cams, anything that needs fast results without needing big hardware.

**Bonus:** After some research, When comparing SSD MobileNet V2 with other object detection models on TensorFlow Hub, you start to see where it really shines and where it falls short. SSD is built for speed and efficiency, especially on devices with low compute power like phones or edge devices. But when you stack it up against Faster R-CNN, the trade-off becomes clear. Faster R-CNN gives you better accuracy and handles busy scenes way better.

EfficientDet is another solid model. It’s scalable, meaning you can pick a version depending on your needs (lightweight or heavy-duty). It hits a nice balance between speed and accuracy, but it’s more complex to set up and train.

YOLO is built for real-time detection. It's super fast and does pretty well on most tasks, but it sometimes struggles with really small objects unless you tweak it right. It’s dope for things like live video or drone footage where speed is key.

As for running more powerful models online like YOLO or Detectron2, the difference is obvious right away. These models give cleaner, tighter bounding boxes and pick up on overlapping or smaller objects that SSD might just ignore. The models are heavier, but they’re better suited for high-stakes use cases like security or medical imaging where every detail counts.