**Brandy Griffin**

**Department of Science, Technology, Engineering & Math, Houston**

**ITAI 1378 Computer Vision**

**Fall 2024**

**Professor Patricia McManus**

**Wednesday, October 30, 2024**

Object Detection using TensorFlow and Pascal VOC

For this assignment, I chose Google Colab with a TPU because it handles heavy calculations for computer vision tasks efficiently. TPUs and GPUs are much faster than standard CPUs, allowing for easier model experimentation. I selected "TPU" in the "Change runtime type" menu and ran code to ensure it was ready. GPUs are great for beginners and perform well for image classification, while TPUs are faster for larger models but may need more coding adjustments. For smaller models like SSD MobileNet V2, a GPU is enough, but TPUs excel with larger datasets. In Colab, the free version has usage limits, so saving work often is important, and using Google Drive for datasets simplifies access.

**Differences**

The main difference between image classification and object detection is that image classification only figures out what objects are in an image, giving the whole image a single label. Object detection, on the other hand, goes further by not just identifying what’s in the image, but also showing where each object is with a box around it.

In our exercise, this difference really stood out. When running object detection, the model didn’t just say there was a person, a car, or a bird in the images—it also drew boxes around each object to show their location. For example, it placed boxes around a person on a horse, a car with its hood up, and a bird, showing that it could find and label each object separately.

**SSD MobileNet V2**

We chose the SSD MobileNet V2 model because it’s a lightweight option that performs well even without a lot of computing power. It balances speed and accuracy, making it ideal for tasks that need quick processing on limited devices, like phones or other small devices. The main advantage is its fast detection and low memory use, but the downside is that it’s less accurate than larger models, especially with complex shapes or smaller objects.

The find\_images\_with\_classes function is useful because it helps filter images with specific categories (like animals or cars), making it easier to focus on target objects in huge datasets like COCO without wasting resources. The threshold=0.5 setting in plot\_detections means only objects detected with at least 50% confidence are shown, so increasing the threshold would show fewer, more certain objects, while lowering it might display more but risk false positives. Heatmaps also help by showing the model’s confidence in certain areas of the image—brighter spots mean the model is more certain an object is there, which helps us see where the model might mix up objects with background details.

In the results, the model performed better at detecting larger, centrally located objects because they’re easier to outline with bounding boxes. It struggled with smaller or partially hidden objects since these can blend into the background. We also noticed that some bounding boxes didn’t fit perfectly around objects, likely due to low resolution or the model’s limits in detecting finer details. If we had used the entire Pascal VOC 2007 dataset, the model's accuracy would probably improve, as it would have more examples to learn from, helping it better locate even less common objects.

To detect specific objects like animals or vehicles, we can tweak the find\_images\_with\_classes function to focus on relevant class IDs. Training a custom model involves gathering and labeling images, setting up a framework like TensorFlow or PyTorch, and preparing the data. We’d also choose a model architecture and train it, facing challenges like obtaining enough labeled data. Despite its limitations, this model is still handy for fast detection tasks, like keeping an eye on video feeds or managing inventory, where getting results quickly is more important than having perfect accuracy.

**Test Yourself**

In this part of the exercise, I used the prompt to display images with detections, but I only saw the same ones as before, featuring a person on a horse, a car with its hood up, and a bird. I also had trouble uploading my own images. While the command worked, I couldn’t get my images to upload, which limited my ability to test the model further.

**Conclusion**

In conclusion, this exercise highlighted the transition from image classification to object detection using the Pascal VOC 2007 dataset with the SSD MobileNet V2 model, which is efficient for limited computational resources. Object detection involves identifying and localizing objects with bounding boxes, aided by the find\_images\_with\_classes function to filter relevant images. The threshold of 0.5 affected the detections displayed. Results showed better accuracy for objects like people and cars compared to birds. While I executed the commands successfully, I couldn't upload my own images which limited my ability to test the model further. Overall, this experience enhanced my understanding of object detection concepts and model evaluation.