Lab: Image Classification vs Object Detection

Devin Prikryl-Martin

Houston Community College

ITAI 1378 Computer Vision

Professor Patricia McManus

October 30, 2024

Lab: Image Classification vs Object Detection

This lab focuses on the functional differences between image classification and object detection, as well as the importance of data size, computational power and pretrained models. The lab begins with installing and importing the necessary libraries, tools and dataset. The dataset in this case is a portion of the PASCAL VOC2007 dataset, which was chosen in order to limit the compute and time required to run the model in this lab. This dataset is a considered a standard benchmark for comparing the performance of object detection algorithms because of its prevalence, the quality and variety of the images it contains and the 20 common object classes it is trained to predict. The pretrained model that will be used is SSD MobileNet V2. This model was chosen because it has relatively undemanding architecture that can be easily loaded from the TensorFlow Hub on devices or platforms like Colab that have limited compute resources. Although it has lower accuracy compared to larger models such as YOLO, sacrificing that accuracy for speed and efficiency, it is still accurate enough to be useful. Especially in cases where the objects are large and the scenes are simple.

The most significant difference between image classification and object detection is that object detection works to not only name the objects in an image, but to draw accurate boundaries around the objects’ location in an image. If the model works well, the output of this exercise will show select images from our dataset, with correctly located and labeled objects. This differs from our previous labs because those displayed only the image labeled by what the model predicted the image contained.

I run into my first issue in code block 5, which I run several times, but continue to get a runtime disconnection at around 25% of “Extraction Complete” while connected to either of the GPU options Colab can use. After trying several times, I switch to TPU to see if it is better able to complete the dataset download. This time it is successful. The first several runs had produced completely inaccurate classification predictions (but decent bounding box predictions) so I hoped to see an improved output. Unfortunately for me, this issue was not the one causing the misclassification. I run the exercise multiple times and struggle to find the source of poor classification. I first get a result of 0 True Positives, 39 False Positives and 331 False Negatives. I will discuss my attempts to produce better results farther on.

The “find\_images\_with\_classes” function is listed as being a fucthion to find images containing the target classes. I don’t see where in the model this function is applied, if at all. It would theoretically be useful on a large dataset because it would allow one to focus on smaller selection of classes. In the “plot\_detections” function, the threshold value of 0.5 only allows detection boxes with confidence scores of over 50% to be displayed, which I’m sure, reduces the number of bounding boxes displayed and increases the likelihood of displaying correct predictions. I cannot get the heatmap visualization to display correctly so I can only guess how it would help. I theorize that the color of the display relates to the model’s confidence about its accuracy.

Despite running the model multiple times and attempting to understand/alter the code, I get the same results or errors no matter what I try. The model can predict with relative accuracy where an object is in space but it can’t seem to produce an accurate classification result. Therefore I cannot describe what sorts of objects it detects more accurately than others as I can not produce an accurate result to begin with. I suspect that the model has a harder time with smaller, detailed, or otherwise visually obscured objects and works best on large, clearly displayed objects. The bounding boxes never seem to miss their targets though, which is something at least. It does tend to see extra objects, when a larger object has many parts. For example, seeing an arm as a separate object from the rest of the body. I expect the accuracy of the model would increase if we used the entire dataset, but I really am not sure as adjusting the percentage of training and validation data used from 10% to 20% showed literally no improvement.

I turned to Claude to ask about what code modifications I could make to detect a specific set of objects. Claude recommends filtering detection results based on class IDs, modifying confidence thresholds for specific classes, or updating the visualization code to only show the desired classes. Although I understand the theory behind Claude’s suggestions, my novice understanding of Python and my limited amount of time to play around with this lab prevents me from being able to practically implement these changes despite my efforts.

If I wanted to train my own object detection model I would need to begin by procuring my data. This could mean using a premade dataset or collecting and annotating my own data from scratch (I’m sure a this would be a laborious, time consuming task.) I would then need to decide on an architecture and framework, probably something in the TensorFlow family since that is where my experience lies. Then the parameters would need to be set, and the model would need to be trained and validated. The challenges I would expect to encounter would firstly be the challenge of my limited knowledge. As I demonstrated to myself in this lab, I am still a weak coder and a IT problem-solving novice. If my technical skills were more up to scruff, I imagine my challenges would be more likely to be related to data, specifically the quantity and quality of labeled data.

Given the limitations of this model, which are in my experience are unfortunately that it does not work, it has very little use in the real world. But if I were savvy enough to get it running, I could see it being useful for a number of applications. Use in monitoring security cams, traffic cams or live wildlife cams for specific objects comes to mind.

Even though I couldn’t get the model to properly function, including being unable to upload and have the model examine my own image, I still learned a lot about object detection and I look forward to progressing my coding skills so that I can actually functionally solve problems under time constraints.

References

Anthropic. (2024, October 30). Claude. “What code modifications could I make to this model that would allow it to detect a specific set of objects?”

Mehta, V. (2021, May 22). *Object detection using SSD mobilenet V2*. Medium. https://vidishmehta204.medium.com/object-detection-using-ssd-mobilenet-v2-7ff3543d738d