Future Computing Technologies Lab: Creative Inquiry

OBJECT DETECTION

April 26, 2021

Bradley Selee
Clemson University
Department of Electrical and Computer Engineering
bselee@clemson.edu

1 Introduction

For my creative inquiry (CI) project, I tried to implement an object detection model into my senior design team's project. The goal of our senior project was to create a handle attachment to a walking cane that helps guide visually impaired people. The idea was to detect medium-large house hold objects using an object detection model, list off the three closest object's as well as its distance and direction using a text-to-speech device and finally guide the user with a vibration system in the handle. This report will focus on the object detection component of our system.

The data I chose to work with was the Common Objects in Context (COCO) data set [2]. It is compiled of 120,000 images with 80 classes that are considered to be common in every day life, the labels are shown in figure 1.1. Each image contains a class ID and coordinates of the object bounding box. Unfortunately, this data set is very unbalanced, highly favoring the outside objects such as road objects and people. This means that there are much more images for these objects, and not as many for the house hold objects. Originally, the plan was to create our own data set, however we quickly realized this was not possible with the allotted time.

```
class_names = ['BG', 'person', 'bicycle', 'car', 'motorcycle', 'airplane',
               'bus', 'train', 'truck', 'boat', 'traffic light',
               'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird',
               'cat', 'dog', 'horse', 'sheep', 'cow', 'elephant', 'bear',
               'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie',
               'suitcase', 'frisbee', 'skis', 'snowboard', 'sports ball',
               'kite', 'baseball bat', 'baseball glove', 'skateboard',
               'surfboard', 'tennis racket', 'bottle', 'wine glass', 'cup',
               'fork', 'knife', 'spoon', 'bowl', 'banana', 'apple',
               'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza',
               'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed',
               'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote',
               'keyboard', 'cell phone', 'microwave', 'oven', 'toaster',
               'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors',
               'teddy bear', 'hair drier', 'toothbrush']
```

Figure 1.1: All classes in the COCO data set

2 Materials and Methods

Due to the complex nature of object detectors architecture and process, this project focuses on implementing an existing model. Because these object detection concepts were very new to me, this project required a lot of research and experimenting with different detectors.

2.1 Research

In general, any model chosen will be trained on the same data set so there could be many different models suitable for the project. Initially, the focus was to understand the history of object detectors, shown in figure 2.1, and use the most advanced one with sufficient documentation. This led to the YOLO and Mask R-CNN object detectors. Looking back, faster R-CNN should have been chosen because the project does not require an object mask, only object bounding boxes.

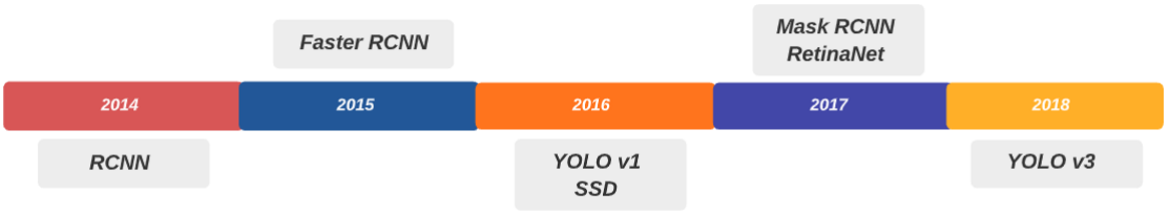

Figure 2.1: Major object detection models

2.2 Initially Choosing a Detector

At first, the YOLO V4 model was explored due to its state of the art technology. While YOLO was probably the best choice, it is much more open sourced than Mask R-CNN leading to less organized documentation. While setting up the YOLO detector, there were a lot of technical errors and data format issues that appeared. YOLO also required a higher knowledge of object detection concepts which made it harder to implement. Due to the knowledge curve of YOLO, and the many different versions, Mask R-CNN was looked into next.

Mask R-CNN was developed by the Facebook AI research team (FAIR) in April 2017 [1]. The main resource for practical application is through a dedicated GitHub repository. This repository has documentation for implementation in different scenarios, training from scratch, and training from prebuilt models. There are also example scripts, jupyter notebooks and a myriad of helper functions for most use cases. For these reasons, Mask R-CNN was the detector most used initially.

2.3 Testing and Practical Implementation

The Mask R-CNN repository contains helpful jupyter notebooks of model implementation as well as examples used in various projects. Typically, these models are pre-trained on large data sets, then tailored to the specific labels intended for detection. The idea behind this is to help the model learn patterns within real objects on large amount of data. This

allows the model to have a starting knowledge when learning other labels. Initially, this is how I learned to implement Mask R-CNN. For practice, the pre-trained weights on the COCO data set were loaded, and the model was fine tuned to detect kangaroos using another data set. With a few inaccuracies, figure 2.2 shows the predictions were mostly correct.

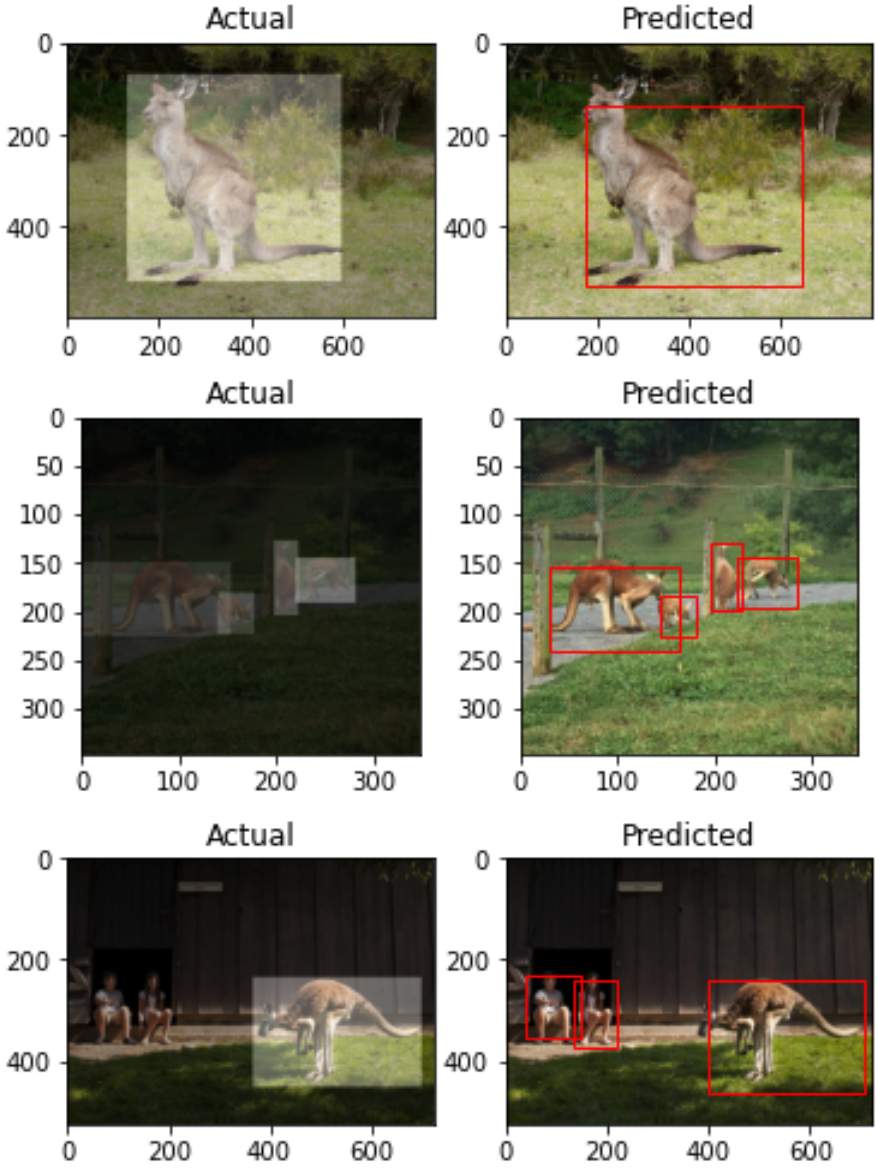

Figure 2.2: Major object detection models

While this was good for practice, the pre-trained weights on the COCO data set already

contained the proper labels for the the project and an extra data set was not needed. After testing the model's inference on images, the next was to run the inference on a live video feed. This is where the model failed. Mask R-CNN has very high accuracy, especially due to the depth of the COCO data set, but very slow inference speed. When testing its real-time detection ability, it only processed a few frames per minute on my local machine. And it would have been nearly impossible on a raspberry pi.

2.4 Final Detector

For the project inference speed was more important than accuracy, therefore a single shot detector was needed rather than a region based detector. After further research, I found MobileNet which was a single shot detector specifically designed for mobile applications, such as a raspberry pi [3]. The documentation for this model was easy to follow and there were already pre-trained weights for the COCO data set specific to this model. Because single shot detectors only need one pass through the image, the speed of detection is much faster than other object detectors. To further improve speed, the floating point numbers in the model were quantized from 32-bit precision to 8-bit precision. The model was immediately used for real-time detection on a live video and performed much better than Mask R-CNN.

3 Results

Results were obtained from both the Mask R-CNN and MobileNet model. The performance was based on how well the model fit for the needs of my senior design project. Due to this, real-time detection was more important than correctly classifying objects every time. The Mask R-CNN detection shows images taken throughout my house and the objects that were successfully detected. The accuracy was very high and it labeled and picked up nearly every object in the frame but, once again, the inference speed was incredibly slow so it was unable to be tested using live video feed in my experience. Figure 3.2 shows a screen shot of the object detection results from a live video feed. This image show about how number of frames per second, the confidence interval of the images detected, and the distance of the object using a depth camera. Some frames were filtered out because they were misclassified, but this was done in the workflow for the senior design project. The main downside is that one can clearly see that the classification accuracy is much lower than with Mask R-CNN.

(a) Living Room

(b) Kitchen

Figure 3.1: Mask R-CNN detection

4 Experience

My overall experience in the CI was gratifying and the continued online class was still very fluent. I really enjoyed everyone screen sharing their progress each week and exploring

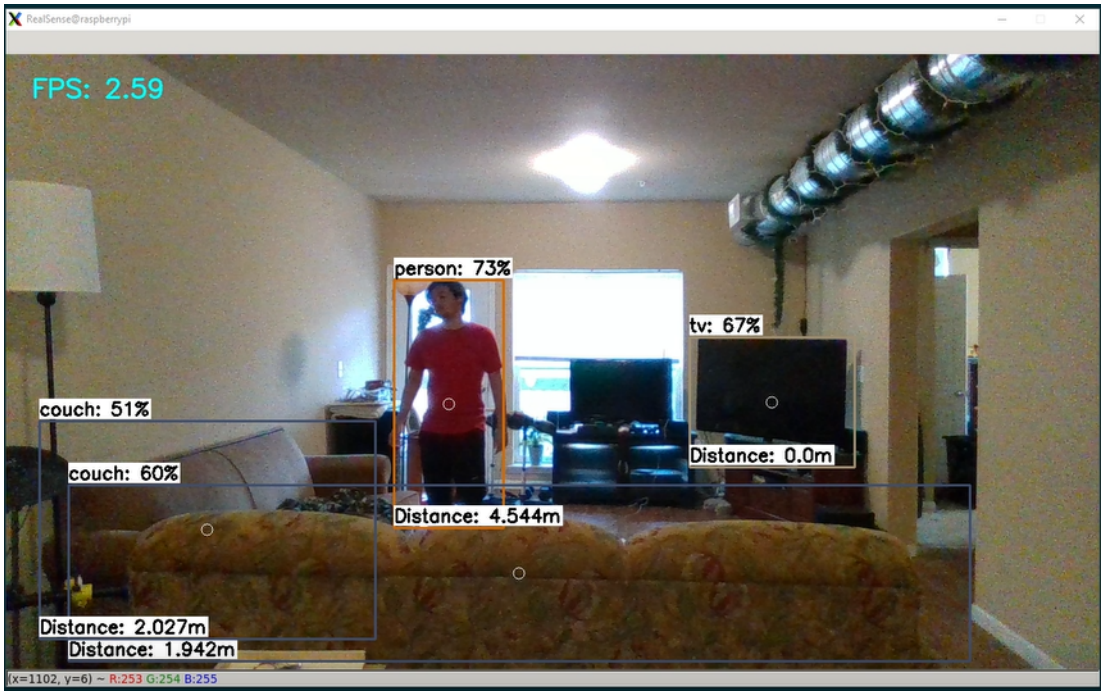

Figure 3.2: Screenshot from MobileNet during real time detection

new methods that they tried. It is hard to say how the CI could be improved due to classes being online, but for continuing students I feel that the CI was managed well. It was very enjoyable having international students and seeing the work they were doing. The main challenge I faced this semester was diving into the research topic of object recognition. I've learned that it is very complex and it is not something that one can become an expert in over a short period of time. The best way I overcame this issue was choosing to work with models that had the easiest documentation to work with. The best resource of this CI is being exposed to practical uses of data science and machine learning, rather than just theory like in traditional classes. I definitely plan to go into some data science field in the future, as this field has recently been exploding and is now more important than ever. I plan to begin my masters with a thesis here at Clemson in Fall 2021. I have reached out to Dr. Smith about graduate research opportunities and it does sound like there will be some.

5 Conclusions and Future Work

It is clear that object detection has come a long way, but there is still much more research to be done. Today, there are exceptional models that have very high classification accuracy when fed large amounts of data. On the downside, these high accuracy models tend to have low inference speed. The same is also true for high speed inference models, while it can process several frames per second, the classification accuracy has a large

decrease. The gold standard of object detection would have high classification accuracy with high inference speed.

Future work on this project would most likely be implementing the YOLO model. While MobileNet did suffice, it seems that YOLO performs better with high classification accuracy. It would be nice to dive deep into the code base of the YOLO model and see how it works from a code standpoint. However, object detection is very complex so it would take a large commitment to understand the code base of the YOLO model.

References

[1] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[2] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. *Lecture Notes in Computer Science*, page 21–37, 2016.