Object Detection using YOLOv5 and VisDrone2019 Dataset

Karl Simon

GW Department of Computer Science

THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

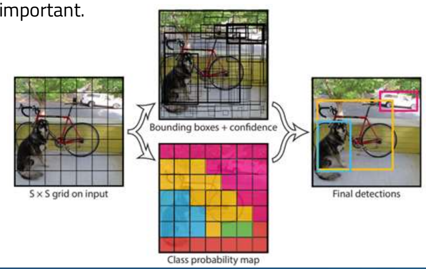
Abstract

As part of a current research project, I am trying to fly a quadcopter autonomously where the drone's position is determined by extracting coordinates from images taken by a mounted monocular camera. In order to accomplish this, the computer on the quadcopter needs to be able to accurately locate targets in the real world.

As this is the first time I have dealt with an object detection task, I was interested in exploring existing machine learning techniques that address this challenge of taking image data (specifically that from a single camera mounted on a quadcopter) and predicting where in the image a desired target appears.

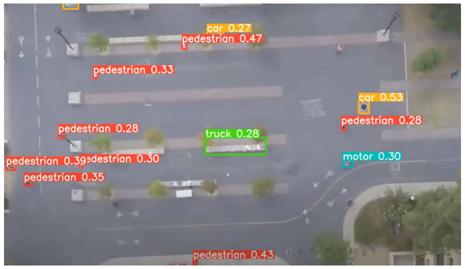
Introduction

Object detection is not a new task, and many algorithms have already been proposed and developed. Typically, these algorithms make use of convolutional neural networks, or CNNs, and other popular examples include Single Shot Detection (SSD) and Faster R-CNN [3]. Of these different approaches, YOLO seemed to be the best for my application because unlike the sliding window and region proposal-based techniques which the other algorithms use, YOLO makes predictions of bounding boxes and class probabilities all at once [2]. This simplified pipeline allows YOLO to be much faster than other models, which for real-time image processing is very important.



Methodology

As a first step in my project, I needed to decide which dataset to use. Specifically, I looked for datasets which contained images taken by drones in interesting environments, and whose objects were already annotated with bounding box coordinates (to save time). Finding a complete dataset was quite challenging, as some were either too large (upwards of 100GB) or didn't have complete annotations.



Still-frame image after using trained model on Stanford Drone Dataset video

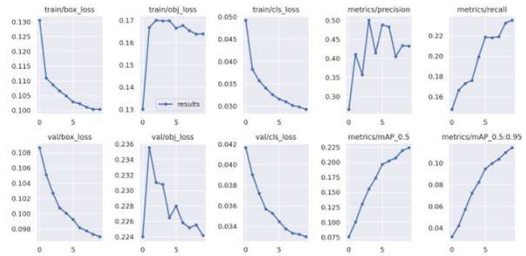
Ultimately, I came across a dataset assembled by a group at Tianjin University in China called VisDrone2019, which included videos and still frames, along with annotations. The benefit of this dataset is that it combines images of common traffic scenes from different angles above the ground, different times of day, and different scales, which means the model could generalize better to test images used from other datasets.

Model Training Information

To train the model on this dataset, I used a Google Colab Tesla T4 GPU, and let it run for 10 epochs. Altogether, preprocessing and training took about 4 hours. Once complete, I used the trained weights of the model to detect test images from the dataset, as well as images from the web and other datasets.

Results

Overall, with the model having to tune over 7 million parameters, the model performed quite well. The mAP metrics increased, which corresponds to more correct positive classifications, and losses decreased, which means IoU error decreased. It should be noted that ideally, the model would be 5 trained for many more epochs to get the most optimal performance. For instance, in viewing the MAP with IoU threshold of 0.5 (bottom right graph below), the curve didn't yet flatten out, so the model was not overfitting yet and results could have been improved.

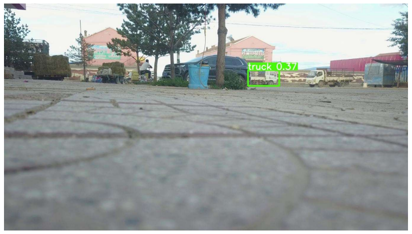


$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

In object detection models, there are two important metrics that determine how accurate a model is on a given dataset; precision and recall; precision refers to the ratio of true positives to all positives, while recall refers to true positives out of all correct predictions

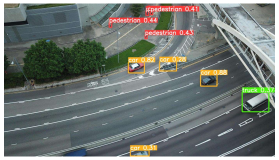


An instance of the model not performing well. Despite there being a truck, vehicle, and pedestrian in the image, they were not detected.

Conclusion

The purpose of this project was to get a first look with how object detection models are constructed and used, specifically using the YOLOv5 model. Although much of the actual implementation of the model was treated as a black box, it gave good insight into how models are evaluated, how to prepare and feed data into the model, and how they can be improved.

An interesting area of future work could be to perform the same task using another model(s), such as R-CNN, and compare it to the YOLOv5 performance. Then the model with the best performance and speed can be adapted for a real-time application. Next, we can use these results along with OpenCV to be able to control a robot based on a camera's image. By calculating the position of the camera relative to the frame of the object being detected, we can determine where the robot is in space, and thus be able to control it.



References

[1] Kiran Krishnan Bonymol Baby Dr. K S Angel Viji Abhinu C G, Aswin P. Multiple object tracking using deep learning with yolo v5. *INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH TECHNOLOGY (IJERT)*, 09 – Issue 13, 2021.

[2] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[3] Divekar A.V. Anilkumar C. et al. Srivastava, S. Comparative analysis of deep learning image detection algorithms. *J Big Data 8, 66*, 09 – Issue 13, 2021.

[4] Fadhlan Zaman, Syahrul Abdullah, Noorfadzli Razak, Juliana Johari, Idnin Pasya, and Khairil Kassim. Visual-based motorcycle detection using you only look once (yolo) deep network. *IOP Conference Series: Materials Science and Engineering*, 1051:012004, 02 2021.