CS4186 Course Project Report

SHYLA KUMAR Rohit
(54581876)

Project Title

Algorithm Implementation and Result Comparison for Object Detection

Group Members

SHYLA KUMAR Rohit (54581876)

Introduction

This project explores different algorithms for object detection that have been developed through the years, interprets and comments on the results obtained. The primary focus of the project is on Single Shot Detectors and Darknet as different methods of object detection. The project shows OpenCV implementations of the above algorithms and compare their results in terms of multiple metrics over the PASCAL - VOC dataset and the time taken to compute their results.

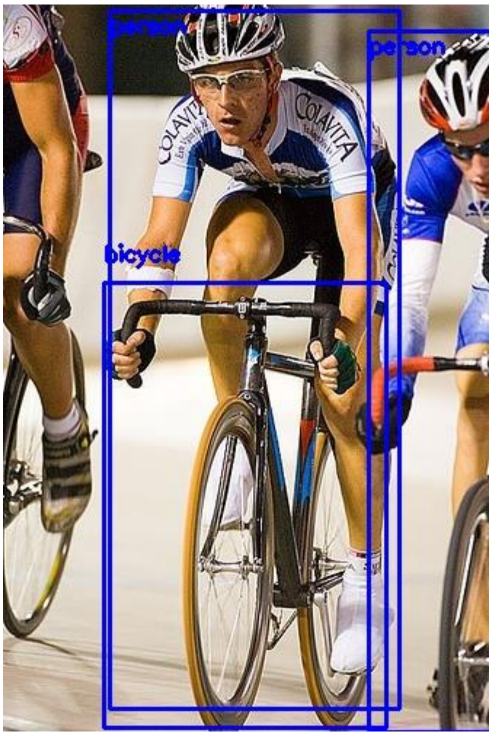
Before we proceed, it is important to define exactly what we're looking for here. Here, an object detector can be thought of as a combination of an object locator and an object recognizer. Hence the algorithms must be able to locate objects in an image and associate them with a pretrained class. This was done in the 20th century, mostly using sliding windows and cascades of classifiers. One popular implementation was the HAAR cascade. I have decided against implementing the HAAR cascade here as it would require extensive training of 20 different cascades for it to perform on the PASCAL – VOC dataset. Instead I have implemented three variants of two popular algorithms that do not use sliding windows at all, instead using one-stage detectors for the task. Both algorithms treat object detection as a regression problem. They are described in greater detail below.

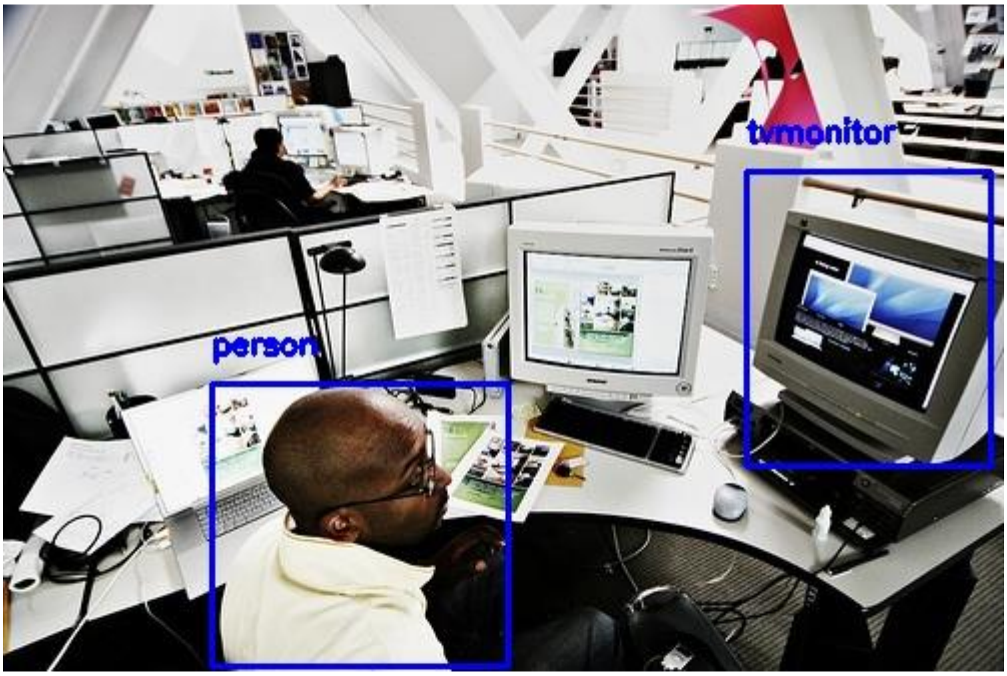
Single Shot Detectors (SSD)

SSDs, originally developed by Google, are a balance between Faster-RCNN models and Darknet YOLO models in terms of speed and accuracy. SSD detection is composed of two parts, extracting feature maps, and applying convolution filters to detect objects.

Step one is computing feature maps which means it computes both the location and class scores using small convolution filters (eg: a 38x38 filter). Step two, after extracting the feature maps, SSD applies 3 × 3 convolution filters for each cell to make predictions. Each filter outputs 25 channels: 21 scores for each class plus one boundary box. Predicting boundary boxes is just like any other Deep Learning problem, we can start with random predictions and use gradient descent to optimize the model.

Results





Metrics

Time taken = 2120.31 seconds

recall = 0.6592

precision = 0.9049

f1 = 0.7628

ratio of objects detected = 0.6592

Darknet

The philosophy of Darknet has always been, You Only Look Once (YOLO). It works by applying a single neural network to the full image.

First, it divides the image into a 13×13 grid of cells. The size of these 169 cells vary depending on the size of the input. For a 416×416 input size that we used in our experiments, the cell size was 32×32. Each cell is then responsible for predicting several boxes in the image. For each bounding box, the network also predicts the confidence that the bounding box encloses an object, and the probability of the enclosed object being a class.

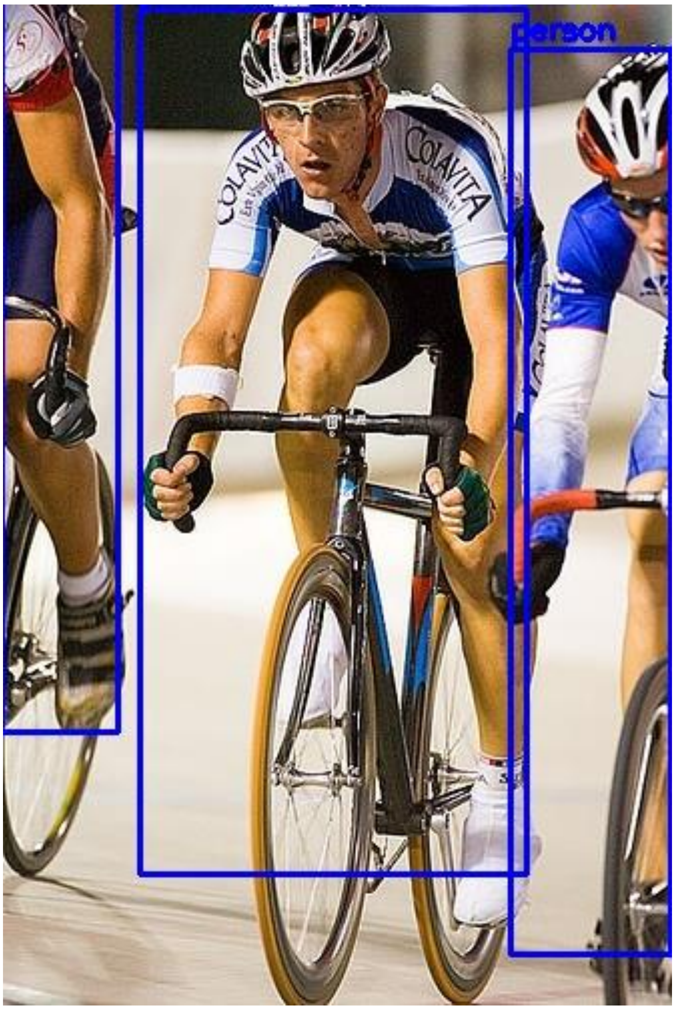
Most of these bounding boxes are eliminated because their confidence is low or because they are enclosing the same object as another bounding box with very high confidence score. This technique is called non-maximum suppression.

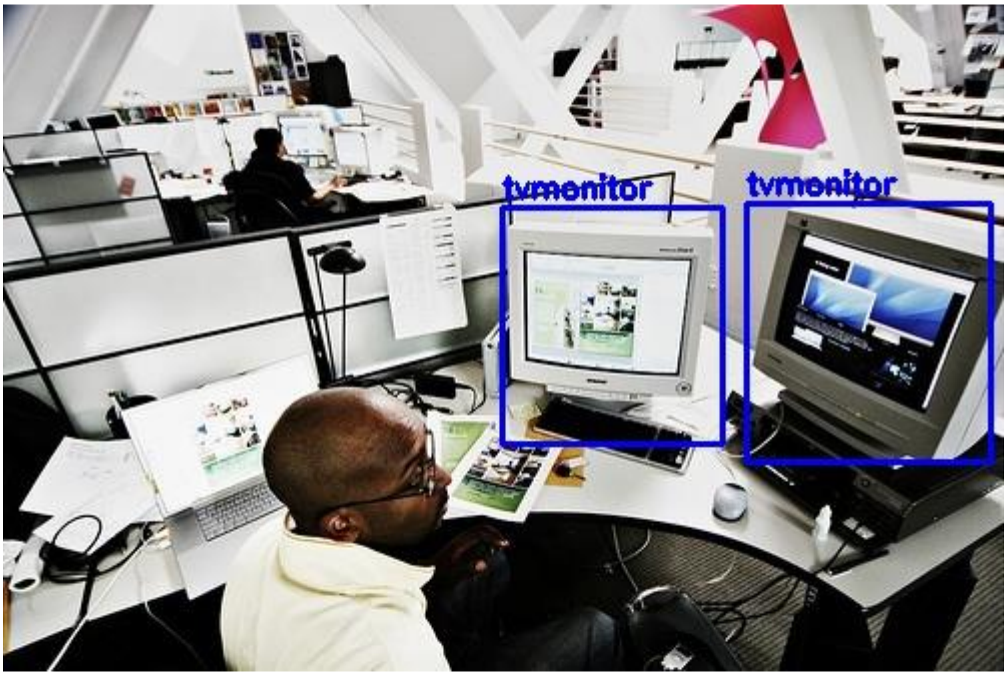
It looks at the whole image at test time, so its predictions are informed by global context in the image. It also makes predictions with a single network evaluation unlike systems like R-CNN which require thousands for a single image.

YOLO v2

This is version two of the YOLO network. It has 24 layers.

Results





Metrics

time taken = 1775.55 seconds

recall = 0.5934175449086877

precision = 0.9479046444064154

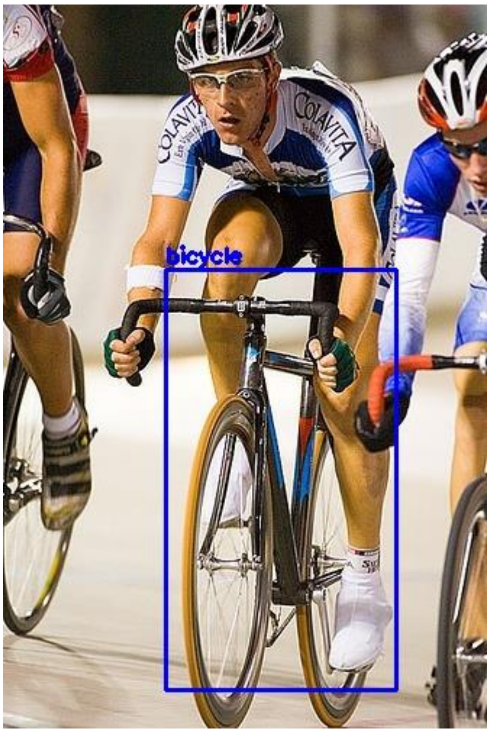
f1 = 0.7298970335866634

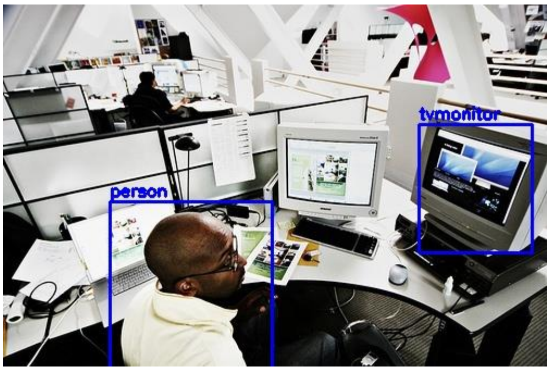
ratio of objects detected = 0.5934175449086877

Tiny-YOLO

Tiny YOLO is a light weight version of the YOLO network and is mostly intended for use on low power devices such as mobile phones. It only has 3 layers.

Results





Metrics

time taken = 628.34 seconds

recall = 0.2930214016991803

precision = 0.950767987065481

f1 = 0.4479783647894566

ratio of objects detected = 0.2930214016991803

Conclusions

Single Shot detectors have the best accuracy of the three models but are also the slowest in terms of time taken. Tiny YOLO appears to be the fastest but weakest when it comes to actually detecting objects and YOLO v2 is a balance between the two.

We can also use these results to see that precision is not a great metric to judge object detection algorithms by. All algorithms have a similar precision because there aren't many false positives generated by object detection algorithms.

In general, algorithms can be improved by increasing the amount of training data and by using deeper networks. Of course, there is a limit to the improvement offered by both these measures, however, in our case, the difference is clearly visible.

Note – Please use the getdata.py script to download the dataset

References

https://arxiv.org/abs/1506.02640

https://arxiv.org/abs/1512.02325

https://arxiv.org/abs/1704.04861

https://pjreddie.com/darknet/yolo/

https://github.com/pjreddie/darknet

https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/