YOLO TensorFlow

Machine Learning

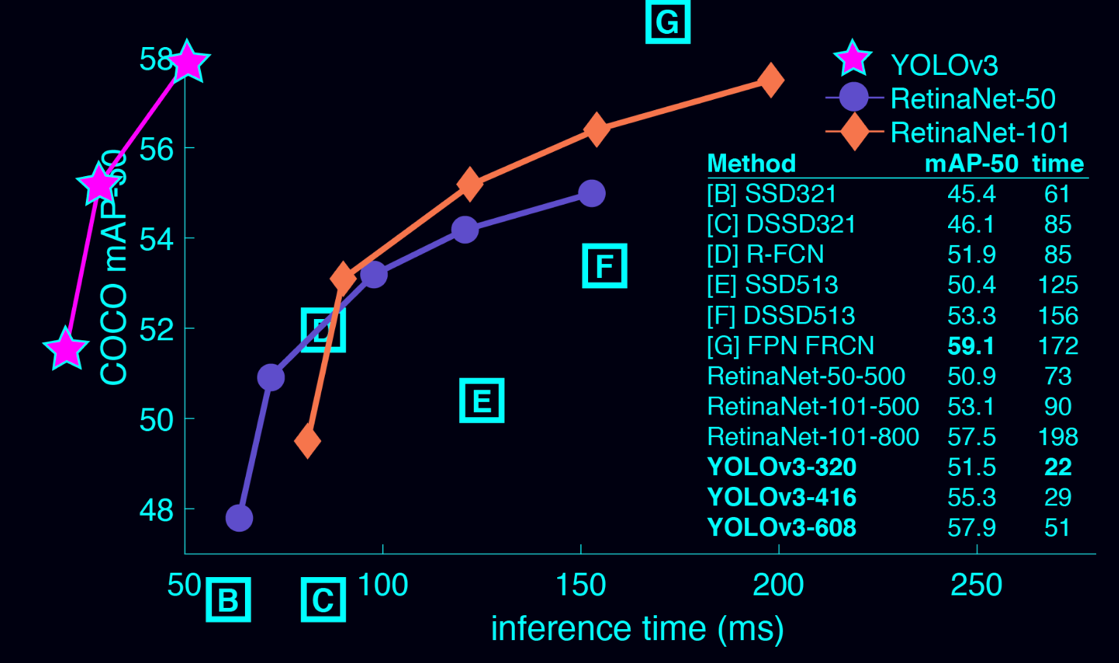
Introduction

Humans glance at an image and instantly know what objects are in the image, where they are, and how they interact. The human visual system is fast and accurate, allowing us to perform complex tasks like driving with little conscious thought. Fast, accurate algorithms for object detection would allow computers to drive cars without specialized sensors, enable assistive devices to convey real-time scene information to human users, and unlock the potential for general purpose, responsive robotic systems.[1]

YOLO is a state-of-the-art, real-time object detection system. On a Pascal Titan X it processes images at 30 FPS and has a mAP of 57.9% on COCO test-dev. [2]It is a new approach to detect objects. It performs a fast and accurate real-time objects detection. Moreover, you can easily tradeoff between speed and accuracy simply by changing the size of the model, no retraining required!

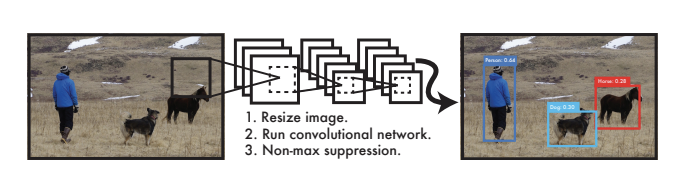
Summary of Reference

YOLO3 is extremely fast and accurate than other algorithms. As image shown below, YOLO3 is x4 faster on par with Focal Loss.



The reason why YOLO is much faster than other algorithm is because YOLO performs object detection in a new way. Prior detection systems repurpose classifiers to perform it at various locations and scales in a test image. Systems like deformable parts models (DPM) use a sliding window approach where the classifier is run at evenly spaced locations over the entire image. [3] More recent approaches like R-CNN use region proposal methods to first generate potential bounding boxes in an image and then run a classifier on these proposed boxes. After classification, post-processing is used to refine the bounding boxes, eliminate duplicate detections, and rescore the boxes based on other objects in the scene [4]. The process for these approaches is pretty slow since each individual component has to trained separately. While YOLO detect straight from whole image pixels to bounding box coordinates and class probabilities and it can be completed in single pipeline. Yolo sees the entire image instead of sliding window and region-propose-based technology during the training and testing.

Processing images using YOLO is pretty simple and straightforward, it can be accomplished by the following steps:



1. Resize the image to 448x448
2. Run the convolutional network on image
3. Threshold the results by module confidence

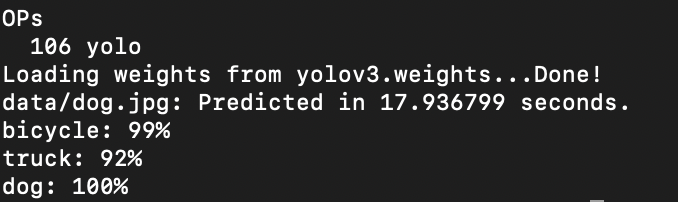
Result

In practice, they connect YOLO to webcam and verify the performance. Pictures below are result for object detection, and the objects could be accurately detected. A demo of the system and the source code can be found on project website: http://pjreddie.com/yolo/.

图片包含 照片, 不同, 展示, 室内

描述已自动生成

If I run detector with one image using YOLO, it takes around 18 seconds to run detection, and the confidence for each objects are displayed. As the data shown above, the detection progress is really fast and the result is accurate. Moreover, the detector is able to work with live stream. See demo: <http://i.imgur.com/EyZZKAA.gif>



图片包含 自行车, 户外, 小狗, 树

描述已自动生成

Pros:

1. YOLO is extremely fast and achieves more than twice the mean average precision of other real-time systems.
2. YOLO makes less background error than other systems.
3. YOLO learns generalizable representations of objects. The detector can test on art-work while train with natural images.

First, YOLO is extremely fast. Since we frame detection as a regression problem we don’t need a complex pipeline. We simply run our neural network on a new image at test time to predict detections. Our base network runs at 45 frames per second with no batch processing on a Titan X GPU and a fast version runs at more than 150 fps. This means we can process streaming video in real-time with less than 25 milliseconds of latency. Furthermore, YOLO achieves more than twice the mean average precision of other real-time systems. For a demo of our system running in real-time on a webcam please see our project webpage: http://pjreddie.com/yolo/.

Second, YOLO reasons globally about the image when making predictions. Unlike sliding window and region proposal-based techniques, YOLO sees the entire image during training and test time so it implicitly encodes contextual information about classes as well as their appearance. Fast R-CNN, a top detection method [14], mistakes background patches in an image for objects because it can’t see the larger context. YOLO makes less than half the number of background errors compared to Fast R-CNN.

Third, YOLO learns generalizable representations of objects. When trained on natural images and tested on artwork, YOLO outperforms top detection methods like DPM and R-CNN by a wide margin. Since YOLO is highly generalizable it is less likely to break down when applied to new domains or unexpected inputs. [1]

Cons:

YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts two boxes and can only have one class. This spatial constraint limits the number of nearby objects that our model can predict. Our model struggles with small objects that appear in groups, such as flocks of birds. Since our model learns to predict bounding boxes from data, it struggles to generalize to objects in new or unusual aspect ratios or configurations. Our model also uses relatively coarse features for predicting bounding boxes since our architecture has multiple down sampling layers from the input image. Finally, while we train on a loss function that approximates detection performance, our loss function treats errors the same in small bounding boxes versus large bounding boxes. A small error in a large box is generally benign but a small error in a small box has a much greater effect on IOU. Our main source of error is incorrect localizations.[1]

Recommendation

YOLO can be used in auto driver or delivery robot since the detector could verify the objects in front of it and it can make decisions to avoid them. It can also be used in real-time reignition to increase the security of restrict area. Yolo provide a fast and accurate technology to detect images so that it could facilitate the facial recognition process.

Reference

[1] <https://arxiv.org/pdf/1506.02640.pdf>

[2] <https://modelzoo.co/model/yolo-tensorflow>

[3] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(9):1627–1645, 2010. 1, 4

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 580–587. IEEE, 2014. 1, 4, 7