International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 9 Issue 8 ǁ 2021 ǁ PP. 16-19

Real-Time Object Detection and Tracking

Dr. Ramya C [1], Nikhil[2], Pavan R V[3], Prabhu nagappa chinagudi[4], Vishal p[5]
[1]Assistant prof, [2,3,4,5]Undergraduate students, Department of Computer Science, MVJCE

Abstract— *Real-time object detection and tracking is a huge and complex field of computer vision. Due to the increase in demand, for tracking the use of security systems, and many other applications, the researchers are forced to constantly improve and improve the performance of more efficient and competitive in the algorithms. However, there are problems in the implementation of the object detection and tracking in real-time tracking in a real-life environment, is expensive, and the calculation is as suitable for the real-time performance, or the tracking of multiple objects using multiple-cameras to complicate this task. Even though many of the practices and the technologies that are available, however, this literature review has covered some of the well-known, and the basic principles and methods of object detection and tracking. Finally, we have passed their terms of application and results of operations*

Date of Submission: 23-07-2021 Date of acceptance: 08-08-2021

I. INTRODUCTION

It is challenging for beginners to distinguish between multiple computer vision tasks. For example, a programmer feels difficulty and confuses between object localization and object detection. But object recognition is process where it combines both object detection and object localization. To gain a complete image understanding, we should not only concentrate on classifying different images, but also try to precisely estimate the concepts and locations of objects contained in each image. This task is referred as object detection object detection is able to provide valuable information for semantic understanding of images and videos, and is related to many applications, including image classification human behavior analysis face recognition and autonomous driving. Meanwhile, inheriting from neural networks and related learning systems, the progress in these fields will develop neural network algorithms, and will also have great impacts on object detection techniques which can be considered as learning systems. However, due to large variations in viewpoints, poses, occlusions and lighting conditions, it's difficult to perfectly accomplish object detection with an additional object localization task. So much attention has been attracted to this field in recent years the problem definition of object detection is to determine where objects are located in a given image (object localization) and which category each object belongs to (object classification). So the pipeline of traditional object detection models can be mainly divided into three stages: informative region selection, feature extraction and classification. We will be using three main tasks in here, those are Image classification -where we predict the class or type of the object present in the image. Object localization -locate the exact position of the object present in the image. Object Detection -here we surround the object in the image with bounding boxes and type of the object located in the image.

II. LITERATURE SURVEY

Paper 1: A real-time object detection algorithm for video
Published year: 23 May 2019

Unlike the fast RCNN, YOLO does not separate the function of seeing an object by multiple processes, such as the region's object prediction and phase prediction. The YOLO algorithm combines both of these functions into a single neural network model to achieve rapid detection with high accuracy. YOLO is an advanced CNN that claims, adopts a convolutional neural network to extract features, and uses a fully integrated layer to predict object details. The model incorporates 24 layers of specification and 2 layers of fully connected. The data sets contain car monitoring videos from smart cities from the Xiamen municipal transport office. That can only distinguish 5 different things. The algorithm is implemented using Python in the TensorFlow framework. What you need is a Nvidia GPU and it depends entirely on the GPU.As the GPU is very expensive, it makes this type of application less effective on standard devices. It uses Fast-YOLO which is a lite version of the algorithm where it faces a problem in the event of a heavy computer load and highly targeted objects.

Paper 2: CPU Based YOLO: A Real Time Object Detection Algorithm
Author: Md. Bahar Ullah, University of Chittagong
Published year: 7 June 2020

Simultaneous forecasting and phase forecasting can be done by YOLO. First, the input image is cut into grid S × S. Second, B-shaped boxes are inserted into each grid cell, each with confidence points. With channels such as Faster R-CNN, Fast R-CNN, etc. However, their frames per second (FPS) on non-GPU computers drive them improperly using the real-time system. The system received some input but the FPS was very low (approximately 2FPS) but higher than that of the black flow. Blob size is a group of pixels connected to an input image or video that provides several standard object-type structures. It was found that the value of the mAP decreased with the size of the web. Blog size depends on computer configuration. The average blob size of CPU based YOLO is not available, which is why the work can be done with better accuracy and FPS.

Paper 3: RefineDetLite: A Lightweight One-stage Object Detection Framework for CPU-only Devices
Author: Chen Chen1 , Meng yuan Liu1 , Xia Ndong Meng2 , Warping Xiao
Published year: 2018

The pyramidal elements are first pulled out to predict the binding boxes and determine whether the anchor is front or rear, called anchor refinement module (ARM). After that, the art object acquisition module (ODM). the input adjustment should match the strength of the spinal network, so we have adjusted the input solution as 320 × 320 for the most efficient spinal cord. Dataset used MSCOCO rain in Pytorch .Used by Intel i7-6700@3.40GHz.The recommended RefineDetLite reaches 26.8 mAP in MSCOCO test-dev at speeds of 130 ms / pic. It only reaches an accuracy of 26.8. Its speed is around 130ms / pic Intel i7-6700@3.40GHz which can provide only 7.69 FPS which is much lower than in real-time mode. It has a very low resolution of 320 * 320.

Paper 4:YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers
Author: Rachel Huang, Jonathan Pedoeem, Cuixian Chen
Published year: 2018.

YOLO (Looks Together) was developed to create a one-step process that involves discovery and division. The YOLO's fast architecture is capable of up to 45 FPS and the smaller version Tiny-YOLO is capable of accomplishing 244 FPS. YOLO-LITE is developing a performance that can handle at least 10 frames per second (FPS) with a non-GPU 30% PASCAL VOC computer. Demonstrates the depth of shallow networks with a non-GPU fast recovery feature. a framework designed to develop YOLO was used to train and test models. Training is done on alienware aura i7 CPU and Nividia 1070 GPU Testing is done on Dell XPS 13 laptop, Struggles to get things closer because each grid can only carry 2 boxes. There is a problem finding small items.

Paper 5: You Only Look Once: Unified, Real-Time Object Detection
Author: Joseph Redmon, Santosh Divvala , Ross Girshick, Ali Farhadi
Published year: 2018

The model is used as CNN and tested on the PASCAL VOC database to find it. The first few layers of network connectivity are used to find features in images and layers that are fully connected. The model layers are followed by 2 fully connected layers.The detection of a small object, such as a flock of birds, is problematic as there is a space limit in the binding boxes. or hundreds of bindings alike.

III. PROPOSED METHODOLOGY

YOLO

Generating a one-step process involves the acquisition of an item and your split Start Look Once (YOLO). YOLO has successfully acquired 45 FPS and is developing a lite version called Tiny-YOLO, reaching approximately 244 FPS (Tiny-YOLOv2) on GPU computers. Why is YOLO different from other existing networks? The answer is: at the same time the binding obligation of the box and the segment prediction can be made by YOLO. First, the input image is cut into S × S grids. Second, B-shaped boxes are inserted into each grid cell, each with confidence points. The potential of each item in the binding box is known as confidence and is defined as:

Our goal is to build a model (CPU Based YOLO) for real-time CPU object detection. YOLO developers have used the Darknet framework to use the algorithm. We are using YOLOv3 with DarkFlow and OpenCv as they are only popular with CPU Based YOLO. Our aim was to improve the model in the Windows operating system but found that installing DarkFlow on windows is a very difficult task. However, we installed YOLOv3 on it, but the result was bad. The FPS was very low and the start time was very large. After that we start doing work on OpenCv. We included the video despite OpenCv and got a good FPS on the output. Art is shown in Fig. We uploaded YOLOv3 and Data (COCO) on OpenCv framework. The upload process process is shown in Fig.

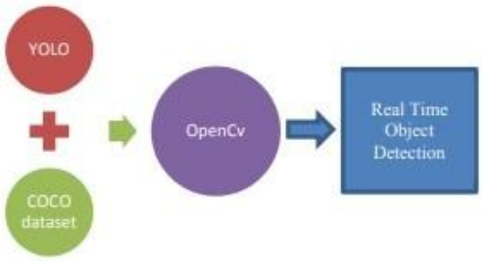
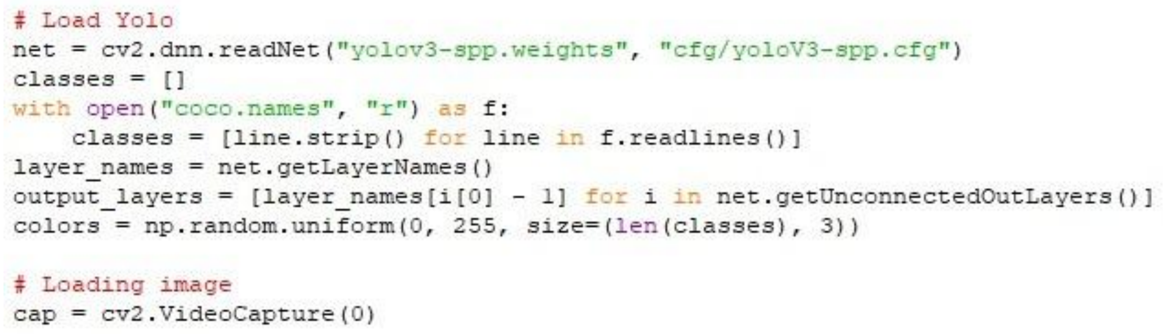


Fig 1. Architecture of CPU Based YOLO

Frames To create our model we need to incorporate an in-depth learning framework in which we will use the YOLO algorithm. There are a few aspects of using the algorithm discussed below.
• TensorFlow: An in-depth learning framework developed by Google that can be used to design, build and train models. But it requires a large amount of GPU power and is attractive for Linux.
• Darknet: Developed by YOLO engineer based on Linux. And it works best on GPU-based computers.
• Darkflow: Made by converting darknet into TensorFlow and works very fast on GPU-based computers. It also works on CPU-based computers but installing windows is awesome and very slow.
• Opencv: Developed by Intel and also has an in-depth learning framework. It only works on the CPU and installation on windows is easy.

When we uploaded the video, we found that the system was getting something input but the FPS was much lower (about 2FPS) but higher than that of DarkFlow. After that we focused on increasing FPS and started reducing the size of the Blob in the input video and saw the performance of getting the test. The upload process is shown in builder 2.

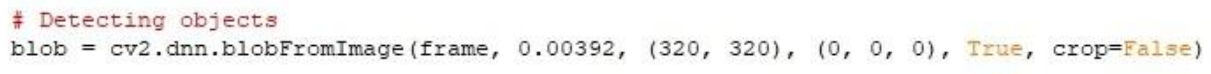


```
# Load Yolo
net = cv2.dnn.readNet("yolov3-spp.weights", "cfg/yoloV3-spp.cfg")
classes = []
with open("coco.names", "r") as f:
    classes = [line.strip() for line in f.readlines()]
layer_names = net.getLayerNames()
output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
colors = np.random.uniform(0, 255, size=(len(classes), 3))

# Loading image
cap = cv2.VideoCapture(0)
```

Fig. 2. Programme of Loading YOLOv3 Using OpenCV

Blob Size Reduction Blob size is a group of pixels connected to an input image or video that provides several common object class properties. We used blob size to get full FPS on CPU-based computers. First, we applied blob 320x320 size to OpenCv function names “cv2.dnn.blobFromImage ()”, obtained 5.5 FPS and 99% to 100% output in webcam video over current algorithms. After that we did the same job several times using webbb size 416x416, 320x320, 128x128, 90x90, 64x64 etc. using several inputs. The computer configuration was AMD Ryzen ™ 5 3600 with 16GB RAM, shown in Fig. 3. We have found that low blob size increases FPS but reduces acquisition accuracy.



```
# Detecting objects
blob = cv2.dnn.blobFromImage(frame, 0.00392, (320, 320), (0, 0, 0), True, crop=False)
```

Fig. 3. Blob Size Reduction Programme

IV. EXPERIMENTAL RESULTS

The same design was used on another device with the AMD Ryzen 3600 CPU configuration, and 16gb ram. Video output was about 5.5 fps, with high accuracy. The CPU was operating at a 60 percent load. By further increasing the CPU load to 100% we have acquired 11 FPS, with high accuracy. Using the LITE version of the algorithm produced at about 30 fps. But the accuracy was low and the camera and object had to be stabilized to get the best performance of the system.

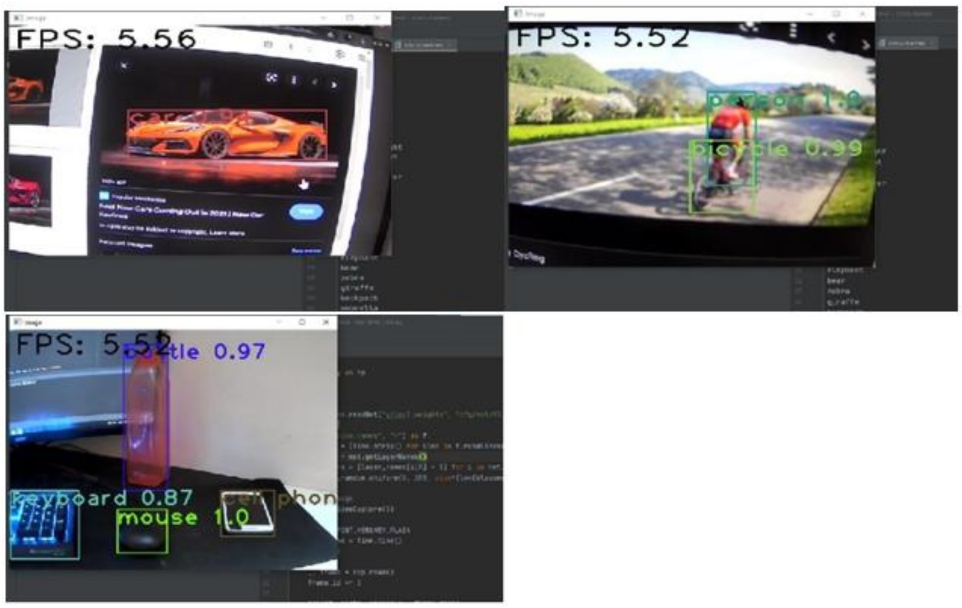


Fig 4 .FPS and Confidence in Different Blob Size at 320*320 from Experiment

V. CONCLUSION

The model is easy to build and can be directly trained in full images. In contrast to the methods designed for segmentation, YOLO is trained for a loss function that is directly related to the acquisition function and each model is jointly trained. YOLO also integrates well into new domains that make it ideal for applications that rely on getting something fast, solid. This makes it an excellent model to choose from for this type of application where speed is important because the products have to be real-time or because the data is very large. Excellent accuracy and excellent speed make YOLOv3 a good model to get something, at least for now. The emerging system is interactive and attractive. While YOLO processes images individually, when attached to a webcam it acts as a tracking system, detecting objects as they change in appearance.

VI. FUTURE WORK

In the future, we hope that this model will be used with the most efficient database of a particular parpose. there are certain limitations that need to be improved in the future. They are as follows:
1) The key to finding scattered objects.
2) It's hard to find small things.

REFERENCES

[1]. Shengyu Lu a , Beizhan Wang a , Hongji Wang a , Lihao Chen b A real-time object detection algorithm for video Elsevier. , pp 398–408, 23 May 2019

[2]. Md Bahar Ullah CPU Based YOLO: A Real Time Object Detection Algorithm.

[3]. Chen Chen, Mengyuan Liu, Xiandong Meng, Wanpeng Xiao, Qi Ju A Lightweight One-Stage Object Detection Framework for CPU-Only Devices. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2020, pp. 700-701

[4]. Mohammad Javad Shafiee, Brendan Chywl, Francis Li, Alexander Wong, Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video

[5]. Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi You Only Look Once: Unified, Real-Time Object Detection.

[6]. Q. Wang and Z. Gao, "Study on a Real-Time Image Object Tracking System," in Computer Science and Computational Technology, 2008. ISCSCT'08. International Symposium on, vol. 2, 2008.

[7]. Y. Meng, "Agent-based reconfigurable architecture for real-time object tracking," Journal of Real-Time Image Processing, vol. 4, no. 4, pp. 339–351, 2009.

[8]. A. J. Lipton, H. Fujiyoshi, and R. S. Patil, "Moving target classification and tracking from real-time video," in Applications of Computer Vision, 1998. WACV'98. Proceedings., Fourth IEEE Workshop on. IEEE, 1998, pp. 8–14

[9]. J. Owens, A. Hunter, E. Fletcher et al., "A fast model-free morphology based object tracking algorithm," 2002.

[10]. C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," International 10)Journal of Computer Vision, vol. 9, no. 2, pp. 137–154, 1992.

[11]. J. Shin, "Initialization of visual object tracker using frame absolute difference," 2000.

[12]. R. Zabih and J. Woodfill, "Non-parametric local transforms for com puting visual correspondence," in Computer VisionECCV'94. Springer, 1994, pp. 151–158.