YOLO based Detection and Classification of Objects in video records

Arka Prava Jana
Department of Telecommunication
R. V. College of Engineering
Bengaluru, India
arkjana98@gmail.com

Abhiraj Biswas
Department of Telecommunication
R. V. College of Engineering
Bengaluru, India
abhirajb9@gmail.com

Mohana
Department of Telecommunication
R. V. College of Engineering
Bengaluru, India
mohana.rvce@gmail.com

Abstract— **The primitive machine learning algorithms that are present break down each problem into small modules and solve them individually. Nowadays requirement of detection algorithm is to work end to end and take less time to compute. Real-time detection and classification of objects from video records provide the foundation for generating many kinds of analytical aspects such as the amount of traffic in a particular area over the years or the total population in an area. In practice, the task usually encounters slow processing of classification and detection or the occurrence of erroneous detection due to the incorporation of small and lightweight datasets. To overcome these issues, YOLO (You Only Look Once) based detection and classification approach (YOLOv2) for improving the computation and processing speed and at the same time efficiently identify the objects in the video records. The classification algorithm creates a bounding box for every class of objects for which it is trained, and generates an annotation describing the particular class of object. The YOLO based detection and classification (YOLOv2) use of GPU (Graphics Processing Unit) to increase the computation speed and processes at 40 frames per second.**

***Keywords*—Object detection, Classification, YOLOv2, video records, performance, object movement**

I. INTRODUCTION

Humans look at an image and instantly process the objects in it and determine their locations due to the interlinked neurons of the brain. The human brain is very accurate in performing complex tasks such as identifying objects of similar attributes, in a very small amount of time. Just like the human interpretation, the world today requires fast and accurate algorithms to classify and detect various objects for various applications. These applications include pedestrian detection, vehicle counting, motion tracking, cancer cell detection and many more [6]. For a human visual system, the perception of visual information is with apparent ease. In artificial intelligence, we face a huge amount of visual information and few useful techniques to process, understand and classify them. Object detection and tracking algorithms are described by extracting the features of image and video for security applications [10] [11]. Features are extracted using convolutional neural network and deep learning [12]. Classifiers are used for image classification and counting [8] [9]. The process of object classification and detection workflow aims to classify objects, based on their features and attributes. As days have gone by, many approaches have been incorporated time to time for obtaining better results. The object detection approaches have progressed from sliding window-based methods to single shot detection frameworks. The Convolutional Neural Network (CNN), in particular, has numerous applications such as facial recognition, as it achieved a large decrease in error rate but at the expense of speed and computation time. The Region based Convolutional Network (R-CNN) uses the process of Selective Search process in order to detect the objects. The descendent of R-CNN, Fast R-CNN and, Faster R-CNN fixed the slow nature of CNN and R-CNN by making the training process end to end [2].

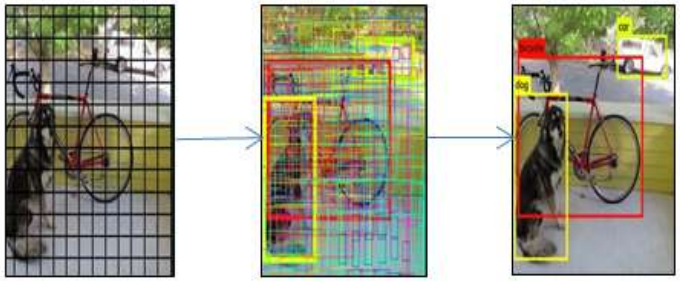


Fig. 1. YOLOv2 model regression.

YOLOv2 (You Only Look Once version 2) is an object detection technique in which the detection process is considered as a single backsliding problem which takes an input image and generates the confidence level of each object in the image[1]. It is the descendent of primitive YOLO algorithm. Fig. 1 depicts a regression model wherein the image which is given as input is divided into grids, followed by formation of bounding boxes on all objects and finally detection of the objects as per requirement. The YOLOv2 detection algorithm finds its genesis to the open source deep learning framework known as Darknet. The Darknet is based on GoogLeNet architecture. YOLOv2 is extremely fast and makes fewer background errors than traditional R-CNN approaches [2]. YOLOv2 divides each image into a several grid boxes and each grid box predicts certain bounding boxes and associated confidence levels. The confidence levels reflect the precision of localization of the objects, regardless of the class. Most of the grids boxes and bounding boxes are removed accounting to fewer threshold values, leaving behind the particular class of objects, which it is trained to detect.

978-1-5386-2440-1/18/$31.00 ©2018 IEEE

Fig. 2. Speed v/s Accuracy of various detection algorithms.

Proposed work is mainly motivated by two main issues that are present in the traditional CNN algorithms. These are low accuracy rate and slow computation speed due to the absence of GPU. Fig. 2 shows the graph of speed versus accuracy of various detection algorithms. This paper, focusing on the working and implementation of YOLOv2 detection algorithm by the YOLO9000 detection system and run it on the video records, which will predict the bounding boxes along with the annotations on the objects. This algorithm is implemented mainly using OpenCV library.

II. DESIGN AND IMPLEMENTATION

This section depicts the overall design requirements and implementation of YOLO model on input images. The section also describes how the model efficiently and accurately detects and classifies objects by implementing Anchor Boxes and CUDA environment.

A. Flow diagram

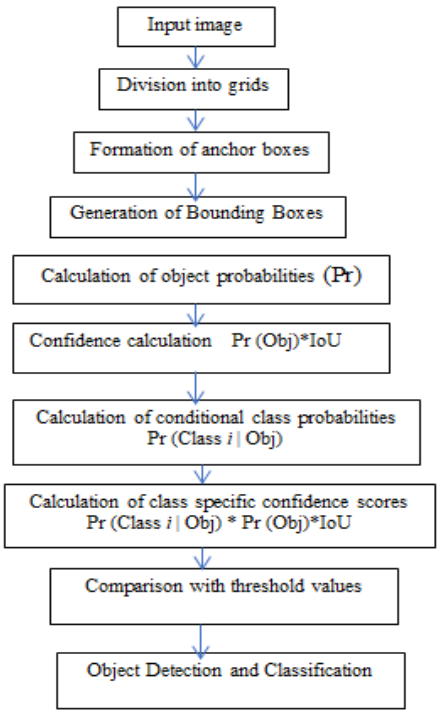


Fig 3. Flow diagram of YOLO Model

Fig 3 shows the flow diagram of YOLO model. YOLO model follows a certain flow method to analyze and detect the objects quickly. Firstly, it follows a regression model wherein it takes the input and derives the class probabilities. Secondly, it calculates the class specific confidence scores. Lastly, it compares the confidence score with the predefined threshold values to detect and classify the objects. If the confidence score is less than the threshold value the algorithm doesn't detect that particular object.

B. Intersection over Union (IoU)

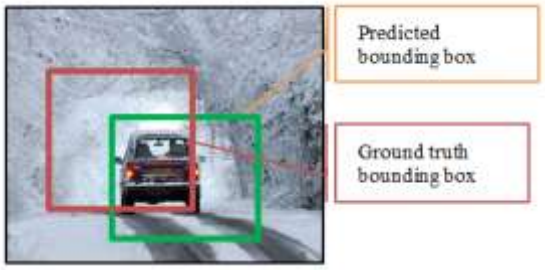


Fig 4. Predicted and ground truth bounding boxes

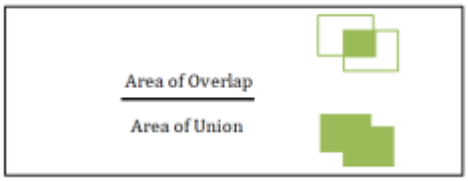
Intersection over Union is a gauging metric that is used to compute the precision of an object detector and classifier on a particular dataset. It consists of two evaluation metrics [7].

The ground truth bounding box: The hand labeled bounding box of a particular object in an image.

The predicted bounding box: The predicted bounding box from the detection and classification algorithm.

Fig. 4 depicts the hand labelled and the predicted bounding boxes. These help in determining the closest bounding box for a particular object.

IoU can also be defined as:



C. Anchor Box

The YOLOv2 model segments the input image into N×N grid cells. Each grid cell has the task of localizing the object if the midpoint of that object falls in a grid cell. But the grid cell approach can predict only a single object at a time. If the midpoint of two objects coincides with each other, the detection algorithm will simply pick any one of the objects. To solve this issue, the concept of Anchor Boxes is used [7].

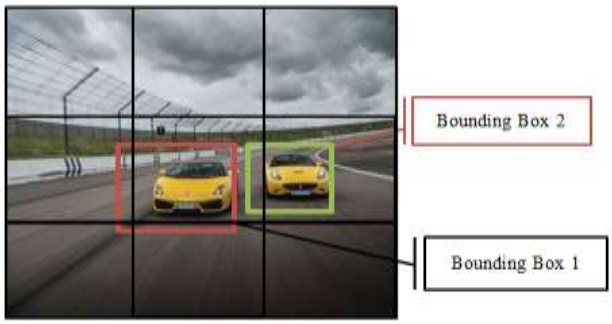


Fig. 5. Generation of grid cells and bounding boxes

In Fig. 5, it is observed that the image consists of two objects (cars) and for ease of explanation chosen N=3 for the number of grids. Here, the image is divided into 3 X 3 grid cells. If the classification and localization algorithm is trained to classify three sets of classes, namely car, person, motorcycle, then the output vector 'Y' of the neural net can be defined as a matrix of 8 possible elements.

$$\begin{bmatrix} P \\ bx \\ by \\ bh \\ bw \\ c1 \\ c2 \\ c3 \end{bmatrix} \quad (1)$$

Equation (1) describes the attributes of an object in an input image. '*P*' defines the presence of object in the grid cell which can take values either 0 or 1, '*bx*' and '*by*' defines the coordinates of the midpoint of the object in a particular grid, '*bh'* defines the percentage height of the bounding box of the total height of the grid cell, '*bw'* defines the percentage width of the total width of the grid cell and *c1*, *c2* and *c3* defines the classes namely person, car and motorcycle. The target volume output will be of order 3 X 3 X 8, where 8 is the number of dimensions defined for this particular classification example.

Consider now a case where two objects share the same midpoint. In this situation, the approach of Anchor Boxes is implemented.

Fig. 6. Implementation of Anchor Boxes

In Fig. 6, it is seen that two objects are sharing the same midpoint. The system of objects now generates an output variable 'Y' as a matrix of 16 elements.

$$Y=\begin{bmatrix} P \\ bx \\ by \\ bh \\ bw \\ c1 \\ c2 \\ c3 \\ --- \\ P \\ bx' \\ by' \\ bh' \\ bw' \\ c1 \\ c2 \\ c3 \end{bmatrix} \quad (2)$$

Equation (2) depicts the possible attributes of the system of objects where the new parameters *bx', by', bh', bw*' are the bounding box parameters of the second object. The ground truth bounding box associated with a particular object is compared with the Anchor Boxes, and the IoU is determined. The object whose IoU metric is maximum will be coded and detected. For instance, in Figure 5 if car is to be detected, then the output variable Y will take the values as seen in (3)

$$Y=\begin{bmatrix} 0 \\ D.C \\ D.C \\ D.C \\ D.C \\ D.C \\ D.C \\ D.C \\ --- \\ 1 \\ bx' \\ by' \\ bh' \\ bw' \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad (3)$$

Don't care

Detection of car

The YOLOv2 algorithm requires a specific set of platforms as it extensively uses the GPU of the system. These platforms readily increase the speed and performance of the algorithm. For a Windows based system, the algorithm requires the help of Microsoft Visual C++ platform.

D. CUDA (Compute Unified Device Architecture)

CUDA is a computing platform created by Nvidia in order to perform general purpose computing on the Graphics Processing Unit (GPU). It works extensively with programming languages like C /C++. The slow speed of a CPU is a serious hindrance to productivity for any image processing computation. CUDA has built-in features that enable it to associate a series of threads to each pixel so that speed is uncompromised. The CUDA environment supports heterogeneous programming that involves a host, that primarily works on the CPU and a device, that consists of the graphics card interfaced with the GPU [4]. The host and the device work hand in hand to improve the workflow and computation speed. The host is responsible for allocating share in memory for the program variables, and the device improves the speed of the computation.In YOLOv2, performance is the main criteria and to achieve a non-dispensable output, CUDA environment plays a vital role. In real time scenario, reduction of noise and redundancy from the objects that are being detected and classified is important [4]. The CUDA environment incorporates a library cuDNN, which provides GPU accelerated functionality in Deep Neural Networks. This environment speeds up the process of smoothening (reduction of noise) and edge detection by manifold due to the associated thread approach and the device-host paradigm.

E. YOLO 9000

YOLO 9000 is a significant improvement to the original object detection system (YOLO). The earlier version of YOLO gave a speed of 35 FPS or 22ms/image. It also was behind in terms of accuracy when compared to other methods like RCNN and

Fast RCNN. There is, therefore, the need to make the original YOLO version better and faster. There is also a need to improve the accuracy. General object detectors pre train on ImageNet dataset on 224*224 and then resize the network to 448*448. Later, they fine-tune the model on detection. This version however trains a bit longer upon resizing before fine-tuning. This increases the mean average precision (mAP) by 3.5%. The earlier YOLO version used the bounding box technique wherein the coordinates of the X, Y, width and height were obtained. This version uses the anchor box technique, which calculates the offsets for images. The offsets are calculated from candidate boxes or the reference points.

YOLO 9000 has also brought the concept of multi scale training into limelight. General object detection systems train at a single aspect ratio (448*448). On the contrary, the new version resizes the network randomly during the training process on a bunch of different scales. The detector is trained on image scales from 320*320 to 608*608. We, therefore, get a network we can resize at test time to a bunch of different sizes and without changing weights that we have trained. We can run the detector at different scales, which gives us a trade-off between performance (speed) and accuracy [5].

There is a dearth of training data in the data detection models. Hence, there is a need for data augmentation. This is also called the Joint training method. The Common Objects in Context (COCO) dataset is used as the detection data set. Although, the COCO dataset can detect multiple objects in an image accurately, however it is confined to only 80 classes. We use the ImageNet as our classification dataset for the model that has 22000 classes. ImageNet, however, labels only one object that is in focus and not on multiple objects in the image. We, therefore, combine the detection and classification datasets and then use backpropagation technique to determine the exact location and class of the image [5]. This result in better performance, more accuracy, and low latency compared to the earlier version of YOLO.

III. SIMULATION RESULTS AND ANALYSIS

For YOLOv2 algorithm to execute and detect the objects, we have employed Microsoft Visual C++ 2017 to build the *.exe* file. We have implemented the pre-trained yolo9000 weights and its configurations. Our system consists of a NVIDIA GEFORCE 940 MX enabled GPU. The Figures in this section depicts the performance of the YOLOv2 model on both still images and on video records. Figure 6 and Figure 8 portrays the detection and labelling of the objects in a single image.

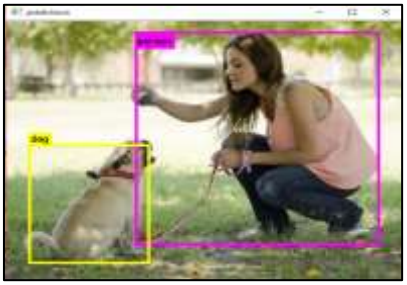


Fig. 7. Detection and labeling of two objects

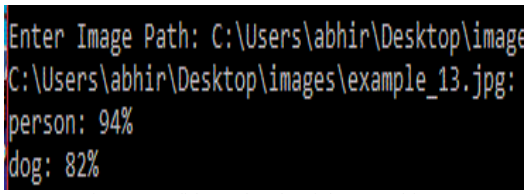


Fig. 8. Confidence levels of the two objects

In Fig. 7, there are two objects and it detects them with comprehensive confidence levels, as shown in Fig. 8. The time for computing the detection algorithm for this image was close to 0.49s.

Fig. 9. Detection and labelling of multiple objects

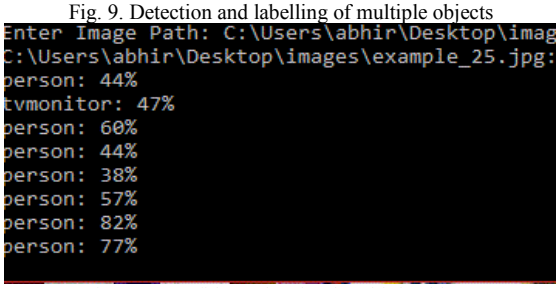


Fig. 10. Confidence levels of multiple objects

Further increase the number of objects in an image the speed of execution doesn't drop. The YOLOv2 model detects majority of the objects with a proficient confidence level. This is portrayed in Fig. 9 and Fig. 10, which has more number of objects compared to Fig. 7. The time for execution for this image was close to 0.5s.

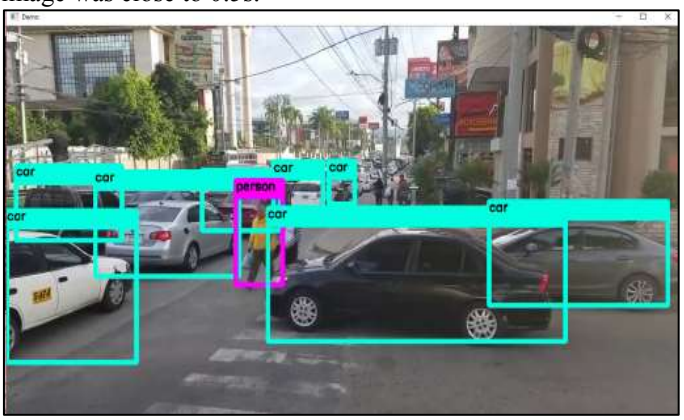


Fig. 11. Detection and Labeling of multiple instances of a single object

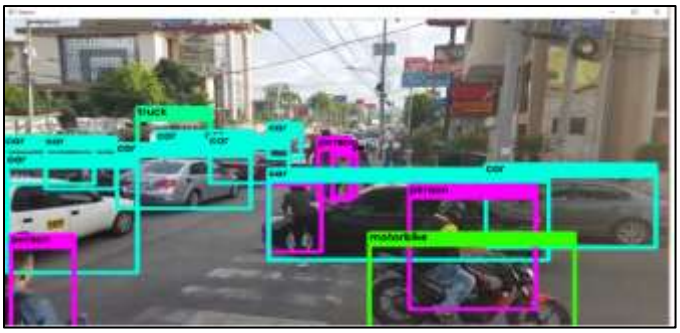


Fig. 12. Detection and Labeling of multiple instances of multiple objects

Fig. 13. Real time object detection and labeling of various objects on a road

As we move from images to video inputs, the scenario completely changes. The objects in a video will now continuously change its co-ordinates. YOLOv2 algorithm here is continuously detect and label the objects with a proficient confidence level. We have taken some still images from the video record that we had given as input. Fig. 11, 12 and 13 depicts the variation in the number of objects in the video. As the number of objects kept increasing, it didn't affect the detection of other neighboring objects. It gives good detection and classification performance.

IV. APPLICATIONS AND CHALLENGES

This section describes some of the applications and challenges.

A. Applications

The fast object detection model, YOLOv2 is used to gain accurate and efficient results. This model is used in pedestrian detection, vehicle detection [3], identifying anomalies in a scene such as explosives, people counting and many more.

B. Challenges

The system requirements for running YOLO model are quite high and it consumes a lot of GPU functionalities to execute. The correct version of CUDA environment (v9.0) plays a key role in setting up the bridge between the algorithm and the GPU. The most challenging part was building the executable file for the YOLO algorithm as it requires many libraries and configuration files to be added.

V. CONCLUSION

In this paper, introduced YOLOv2 model and YOLO9000, real-time detection systems for detecting and classifying objects in video records. YOLOv2 is agile and efficient in detecting and classifying the objects. The speed and accuracy were achieved with the aid of GPU functionalities and Anchor Box technique respectively. Furthermore, YOLOv2 can detect object movement in video records with a proficient accuracy. YOLO9000 is a real-time framework which is able to optimize detection and classification and bridge the gap between them. The YOLOv2 model and YOLO 9000 detection system collectively are able to detect and classify objects varying from multiple instances of single objects to multiple instances of multiple objects.

REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You Only Look Once: Unified Real-Time Object Detection", *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 779-788, 2016.

[2] Ren S, He K, Girshick R, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.", *2017 IEEE Transactions on Pattern Analysis & Machine Intelligence.*

[3] Jing Tao, Hongbo Wang, Xinyu Zhang, Xiaoyu Li; "An object detection system based on YOLO in traffic scene" *2017 6th International Conference on Computer Science and Network Technology (ICCSNT).*

[4] Shunji Funasaka, Koji Nakano, Yasuaki Ito, "Single Kernel Soft Synchronization Technique for Task Arrays on CUDA-enabled GPUs, with Applications", *2017 Fifth International Symposium on Computing and Networking (CANDAR)*

[5] Joseph Redmon, Ali Farhadi, "YOLO9000 Better, Faster, Stronger", *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

[6] Ipek Baris, Yalin Bastanlar, "Classification and tracking of traffic scene objects with hybrid camera systems", *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC).*

[7] Arsalan Mousavian, Dragomir Anguelov, John Flynn, Jana Košecká, "3D Bounding Box Estimation Using Deep Learning and Geometry", *2017 IEEE Conference on on Computer Vision and Pattern Recognition (CVPR).*

[8] Mohana and H. V. R. Aradhya, "Elegant and efficient algorithms for real time object detection, counting and classification for video surveillance applications from single fixed camera," *2016 International Conference on Circuits, Controls, Communications and Computing (I4C)*, Bangalore, 2016, pp. 1-7.

[9] H. V. Ravish Aradhya, Mohana and Kiran Anil Chikodi, "Real time objects detection and positioning in multiple regions using single fixed camera view for video surveillance applications," *2015 International Conference on Electrical, Electronics, Signals, Communication and Optimization (EESCO)*, Visakhapatnam, 2015, pp. 1-6.

[10] Akshay Mangawati, Mohana, Mohammed Leesan, H. V. Ravish Aradhya, "Object Tracking Algorithms for video survilllance applications" *International conference on communication and signal processing (ICCSP),* India, 2018, pp. 0676-0680.

[11] Apoorva Raghunandan, Mohana, Pakala Raghav and H. V. Ravish Aradhya,"Object Detection Algorithms for video survilllance applications" *International conference on communication and signal processing (ICCSP),* India, 2018, pp. 0570-0575.

[12] Manjunath Jogin, Mohana, "Feature extraction using Convolution Neural Networks (CNN) and Deep Learning" *2018 IEEE International Conference On Recent Trends In Electronics Information Communication Technology,(RTEICT)* 2018, India.