TEST Engineering & Management

May - June 2020
ISSN: 0193-4120 Page No. 4911-4914

Object Detection using Deep Learning Techniques

[1]Pratyush Singh, [2]Rithik Kumar Jha D, [3]MD Faishal Khan, [4]Rohit Raj, [5]K. Amuthabala

[5]Assistant Professor, [1,2,3,4,5]School of C&IT, REVA University, Bengaluru, KA
[1]tops@yahoo.in, [2]rithikkumar53@gmail.com, [3]khanfaishal987@gmail.com,
[4]rohitraj13may1998@gmail.com, [5]amuthabala.p@reva.edu.in

Article Info
Volume 83
Page Number: 4911-4914
Publication Issue:
May - June 2020

Article History
Article Received:*19 November 2019*
Revised: *27 January 2020*
Accepted: *24 February 2020*
Publication: *16 May 2020*

Abstract
This Object Detection Project deals with the identifying of the object in an image (Or Video which is just a bunch of Moving images) provided as an input. Object detection has multiple applications such as face detection, vehicle detection, pedestrian counting, self-driving cars, security systems, etc. The application will be using Tensorflow along with Keras developed by Google. The base language used for development and deployment is Python and C++ tested on Linux. The algorithm to be used in our project for Object detection and classification is YOLOv3 using Nvidia's CUDA. YOLO is an accurate model which divides images into regions and predicts bounding boxes and probabilities for each region.

Keywords: *Interaction under union (IOU), You look only once (YOLO), deformable parts models (DPMs), Probability (Pr), convolutional neural network (CNN)*

1. Introduction

The natural eye is able to do remarkably distinguishing various articles in an image quick and precisely making us proficient to perform complex errands like driving. Quick and exact, calculations for object recognition will permit machines/PCs to hold out complex undertakings without particular sensors, empower assistive gadgets to pass on continuous scene data to the clients. To detect an object, these systems take a classifier for that object and assess it at different areas and scales in a very test image. Systems like deformable parts models (DPM) use a window approach where the classifier is run at evenly spaced locations over the complete image. newer approaches like R-CNN algorithm use region proposal methods to first generate potential bounding boxes in a picture then run a classifier on these proposed boxes. After classification, post-processing is done to polish the bounding box(s), destroy copy detections, and re-asses the scores the box supported different objects inside the scene. We re-outline object identification as one relapse issue, directly from picture pixels to bounding box directions and modernity probabilities. Using the system, you process a image only once to predict what objects are present and where they're present. YOLO is quite simple: one convolutional network parallelly predicts more than one bounding boxes and sophistication probabilities for those boxes. The algorithm which we are implementing, YOLO utilizes full pictures for preparing and enhances execution as needs be. This sort of brought together model has some useful advantages over customary method(s) for object identification. Subsequently, we chose to move on with YOLOv3 calculation which is actualized in our undertaking.

2. Literature survey

To approach the matter of object detection using neural networks and by the introduction of Regions with R-CNN, regions are basically used for object detection. Faster R-CNN which takes an extra subnetwork to come up with region proposals.[4] Shortcomings of R-CNN are that in spite of the fact that it utilizes particular search and other preprocessing steps to extricate the potential bounding box for input, R-CNN still incorporates a genuine speed bottleneck - the rehashed calculations will happen in light of the fact that the PC finds the highlights for each locale. This builds the time required for the model to figure on each picture which impedes its certifiable applications.[4]

TensorFlow utilizes a brought together dataflow diagram to appeal to both the calculation in a calculation and in this manner the state on which the calculation

TEST Engineering & Management

May - June 2020
ISSN: 0193-4120 Page No. 4911-4914

works. Tenserflow Model is made utilizing information inputs which experiences preparing with various shrouded layers. [1]

A tensor consists of a group of primitive values shaped into an array of any number of dimensions. Tensor a mathematical object analogous to but more general than a vector, represented by an array of components that are functions of the coordinates of an area. [2]

TensorFlow uses one dataflow graph to represent all computation and state in a very machine learning algorithm, including the individual mathematical operations, the parameters and their update rules, and therefore the input pre-processing. [2]

Currently, the sole supported GPUs are that of NVIDIA and therefore the only full language support is of Python which makes it a downside as there's an increase of other languages in deep learning. it's very hard and complicated to know which makes it even harder to figure with.[3]

Now coming to the very first version of YOLO. one convolutional network simultaneously predicts multiple bounding boxes and sophistication probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance. [5]

Drawbacks are handling Groups of small objects, Unusual aspect ratios struggles to generalize, Coarse Features (Due to multiple pooling layers from input images), Localization error of bounding boxes.

The problems faced by the previous authors are quite interesting, as usually neural networks, convolutional layers, and deep learnings are complex topics but with further increase in complexity while implementation possesses greater challenges for the developers. this can be one among the issues we are attempting to resolve with YOLOv3 approach to solving object detection problem(s). In most of the algorithms previously known for object detection are quite slow and uses lots of resources, including special requirements like CUDA cores by Nvidia, OpenGL, etc together with this one input image is processed again and again before the result's achieved. this can be where our approach is healthier than the legacy algorithms, YOLO uses one image, just once to detect objects and to predict bounding boxes, this method solves the age old problem of object detection being a resource hungry systems and makes it more accessible and usable for large/small scale systems. Unlike older algorithms our approach is easy, fast, less resource hungry and simple to implement & understand.

3. Methodology

A. Bounding Box Forecasting

YOLOv3 uses a set of dimensions to provide anchor frames. The framework predicts bounding boxes utilizing dimension clusters as anchor boxes. Utilizing the calculated relapse where a procedure covers the decision square shape first on the object of the basic truth, YOLOv3 predicts the objectivity.

This compensates for one bounding box before a solitary article and any blunder during this would happen both inside the assignments and inside the recognition sum (objectivity). Other than being different precursors of the decision square shape that may have an objectivity score over the verge yet under the most effective/best, these blunders happen only for the discovery shortfall yet not for the distribution reason. The system utilized in this task predicts 4 directions for each bounding box, that is tx,ty,tw,th.[5]

$$b_x = \sigma(t_x) + c_x$$
$$b_y = \sigma(t_y) + c_y$$
$$b_w = p_w e^{t_w}$$
$$b_h = p_h e^{t_h}$$

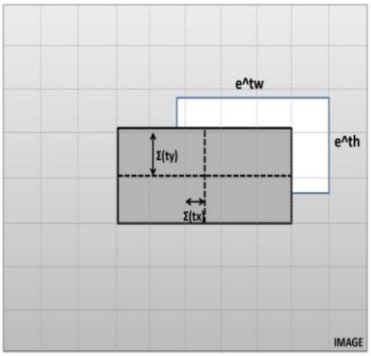

Figure 1: Bounding Box Prediction

B. Bounding Box Prediction

YOLO makes a solitary substance for every one of the different segments of object detection into one neural system. The system utilizes highlights from the whole picture to anticipate every single bounding box. It additionally predicts each bounding boxes for an image at the same time. This infers that the system looks all inclusively about the whole picture and every object(s) inside a picture. The YOLOv3 configuration empowers start to finish preparing and continuous rates while keeping up high normal exactness. The system separates the picture to a S X S matrix. In the event that the focal point(s) of an object falls into a matrix cell(out of clusters of lattice cells), that framework cell is answerable for identifying that class thing or we can just say an object. we'll be utilizing k-grouping to work out the bounding box priors. Every matrix cell predicts B bounding boxes long with the certainty scores for the boxes anticipated by the system. These certainty scores reflect how sure and the model is that the box contains an object and furthermore how precise it thinks the box is that it had anticipated. The confidence is defined as:

May - June 2020
ISSN: 0193-4120 Page No. 4911-4914

$$Pr\ (Object) * IOU^{truth}_{Predicted}$$

If no object is found to be in that cell, the confidence scores should be zero. IOU between the predicted box and the ground truth should be equal to the confidence score.x,y,w,h, are five predictions present in the bounding box along with the confidence. The (x,y) coordinates represent the center of the box relative to the boundaries of the relevant grid cell. Relative to the whole image, width and height is then predicted. Finally, the confidence prediction depicts each cell from cluster of grid cells along with predicting conditional class probabilities(pr),

$$Pr\ (Class_i|Object)$$

These probabilities mentioned are based on the conditions on the grid cell which is containing an object. Single set of class probabilities are predicted per grid cells, no matter what no. of boxes have been depicted by B

At test time we multiply the individual box confidence predictions with the conditional class probabilities

$$Pr\ (Class_i|Object) * Pr\ (Object) * IOU^{Truth}_{Pred} = Pr\ (Class_i) * IOU^{Truth}_{Pred}$$

This gives us class-specific confidence score(s) for each box. These scores include the probability of that class shown in the box.

C. Class Prediction

Each box predicts the classes, the bounding box may contain utilizing multilabel classification. УOLOv3 utilizes a various classification by tag. Therefore, УOLO executes a soft-max ability to make an understanding of the scores into probabilities that signify in any event one. УOLOv3 changes the soft-max work with individualistic vital classifiers to decide the probability that the thing has a spot with a label. as opposed to utilizing the mean square blunder to determine the classification misfortune, УOLOv3 utilizes the double loss of cross entropy for each label. We don't utilize a SoftMax utilizing a SoftMax forces the conviction that each box has precisely one class which is generally not the situation. During preparing we utilize parallel cross-entropy misfortune for the class forecasts. This lessens the multifaceted nature of the estimation by maintaining a strategic distance from the soft-max work. This detailing causes after we move to increasingly complex spaces simply like the Open Images Dataset. during this dataset there are many covering labels.

D. Feature Extractor

The approach used in the network can be called ass a mixture between the neural network(s) utilized in УOLOv2 and Darknet-19.[2] It uses successive 3*3 and 1*1 convolutional layers. Darknet-53 is that the third scope of parts from layer 0 to layer 74, there are 53 convolutional layers and thusly the rest of the levels are said to be inhabitant layers, much the same as the fundamental system structure for the extraction of УOLOv3 characteristics. These convolutional layers are gained by fusing convolutional layers with extraordinary presentations of shifted customary framework structures.

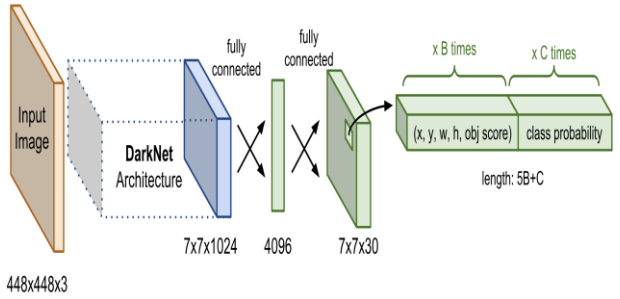

Figure 2: Network Architecture

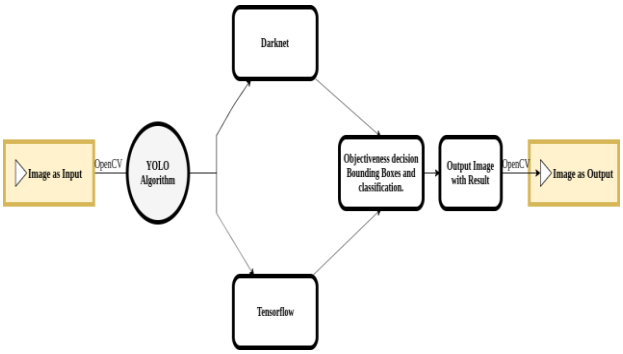

Figure 3: Data flow graph

4. Implementation

System Requirements and Dependencies

- x64 CPU | Intel i3 + | AMD
- Python 3.x with OpenCV
- Darknet Dataset
- Numpy
- Nvidia CUDA GPU Recommended for training custom weights.

For training the model we have used COCO Dataset from Darknet, the training was done using OpenCV on CPU at 14 FPS. The following COCO dataset is then singled into one file with extensions *. weights.

This Trained weight is stacked, at that point we start by furnishing the algorithm with a picture on which deduction must be done, at that point we will be parting the picture into S×S cell(s). The cell is held at risk for distinguishing the item on the off chance that it falls there under particular cell.

Each cell predicts

(a) The placement of B, bounding boxes

(b) A Confidence Score.

(c) A likelihood of item class adapted on the presence of an article inside the bounding box.

TEST Engineering & Management

May - June 2020
ISSN: 0193-4120 Page No. 4911-4914

Tuple of 4 values, (center x-cord, center y-cord, width, height) — (x,y,w,h) shows the coordinates of bounding box, where x and y are set to be offset of a cell location.

Moreover, x, y, w and h are standardized by the picture width and tallness, and in this manner all between (0, 1].

A confidence score indicates the likelihood that the cell contains an object:

$$Probability(containing\ an\ article)\ x\ IoU^{Pred}_{Truth}$$

It predicts a probability of this object belonging to every class Ci,i=1,…,K: (If the cell contains an object)

$$Probability(the\ article\ belongs\ to\ the\ class\_C[i]\ |\ containing\_an\_object).$$

At this stage, the model just predicts one lot of class probabilities per cell, paying little mind to the quantity of bounding boxes, B.

Altogether, one picture contains S×S×B bounding boxes, each box similar to 4 area forecasts, 1 certainty score, and K contingent probabilities for object classification. The all out forecast esteems for one picture is S×S×(5B+K), which is the tensor state of the last convolutional layer of the model utilized.

The final layer is modified to output a prediction tensor of size

$$S \times S \times (5B + K).$$

5. Results and discussions

Object detection was achieved and implemented using the advance YOLOv3 algorithm, it helped us understand the working of convolutional layers and deep neural networks. Through practical implementation, we can say that the YOLOv3 algorithm has progressed a lot from its first version and is a lot faster and better. After the training, the classifier achieves a top accuracy of 93.3%,

$$Accuracy = TP / TP + FP$$

TP- True Positive, FP- False Positive

We were able to deploy the system developed under this project with minimal hardware and software dependencies, this proves that the network is very light on system resources and is very fast to detect objects of interest. We were quite satisfied by using this method for object detection due to agility and simplicity to implement.

6. Conclusion

The Project "Object Detection Using Deep Learning" is an approach to understand Machine learning The algorithm(s) which we have used to implement are State of the art machine learning Algorithms that are very different and more accurate than Classical Machine learning Techniques, which are more based on Human Perception on Identifying Patterns. While referring to the previous research material it is found that the problems pertaining to older algorithms was that they were too complex and have difficult implementations and most of all these older traditional algorithms processes a single (input) image multiple of times but the network we are using only processes the image once. One of the goals is to show how Deep Learning has Progressed and can be used on different types of applications for Object Detection and Classification of detected objects into meaningful data. The YOLOv3 algorithm is designed to work with great accuracy in Training of Prediction Models and to work with pre-trained models when compared to other older approaches to the problem of object detection and solves the challenges faced by legacy algorithms/methods of Object detection.

7. Acknowledgment

We would like to thank our college, REVA University for providing us an opportunity to present our idea as a project. We would also like to express our gratitude to the Director of our department, Dr. Sunilkumar Manvi for inspiring us to come up with innovative and deviceful propositions. We would also like to take this opportunity to thank our guide, Prof. Rajesh I.S who has been there with us in every step and has helped us in every way possible. Lastly, we would like to extend our thanks to Prof. K. Amuthabala who has supported us and mentored us in every venture of ours.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, "TensorFlow: A System for Large-Scale Machine Learning", 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '17)

[2] Mohd Azlan Abu, Nurul Hazirah Indra, Abdul Halim Abd Rahman, Nor Amalia Sapiee and Izanoordina Ahmad, "A study on Image Classification based on Deep Learning and Tensorflow"- 2018

[3] Joseph Redmon, Ali Farhadi, "YOLOv3: An Incremental Improvement", University of Washington – 2018

[4] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu, "Object Detection with Deep Learning: A Review", IEEE 2019

[5] Geethapriya. S, N. Duraimurugan, S.P. Chokkalingam, "Real-Time Object Detection with YOLO", International Journal of Engineering and Advanced Technology (IJEAT) 2019.