International Journal of Scientific Research & Engineering Trends
Volume 7, Issue 1, Jan-Feb-2021, ISSN (Online): 2395-566X

A Survey of Real Time Deep Learning Based Object Detection

Ahmad Ismail
Department: Informatics and control systems IU
Bauman Moscow state technical university (BMSTU)
Moscow

***Abstract*- The importance of artificial intelligence manifests in all of our life's aspects, robotics, self-driving, surveillance, agriculture, transportation, medicine, space, and military. Deep learning is the paramount core for AI applications; especially object detection, localization, and tracking. The convolutional neural networks make momentum regarding the aforementioned topics. The architecture in this field varied from the fine-grained to the real-time models. Meanwhile, the fine grained models produce the maximum possible accuracy; it consumes time on the other side, real-time models are featured with speed, but it provides less accuracy.In this paper, we will review start-of-the-art models being used for near real-time object detection and tracking, then introduce our approach; an accurate real-time object detector and tracker.**

***Keywords*-Deep Learning, Object detection and tracking, Real time system**

I.INTRODUCTION

The demand for real-time embedded systems increased in many applications including, industry, medicine, and avionics. Suppose the on-board computing unit in an autonomous military airplane receives the images from the camera; the software running in the CPU must perform all required operations to detect and track the target (besides the guidance and control) in real-time.

The state-of-art real-time embedded hardware was developed to meet the requirements; Sabir Hossain and Deok-jin Lee[1] mentioned two categories of real-time embedded modules; they tested object detection and tracking over different modules using many deep learning-based algorithms.They concluded with YOLOv2 (you look only once) [2] is faster and accurate, but the detecting algorithm is limited by 20 m.

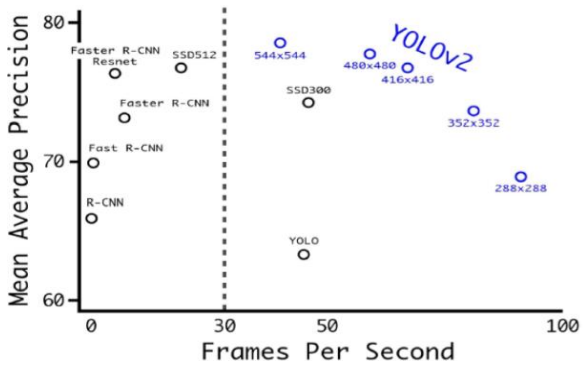

Fig 1. YOLOv2 performance comparison with other models trained on PASCAL VOC 2007.

This model provides 76.8 mAP [3] trained on 544x544 images with 40 FPS, which assessed fast enough, but the accuracy 76.8% is not enough. YOLOv2 attained better performance regarding other models, especially with higher frame rates.

Faster R-CNN [4] and SSD512 [5] made improvements in deep learning-based object detection in the last decade while YOLOv2 represents a leap for real-time system object detection and tracking in speed and accuracy; this paper answers the two main questions:

- What makes YOLOv2 so effective and efficient in both accuracy and speed?
- How can we improve the accuracy in this model for a specific frame rate?

II. FASTER R-CNN ARCHITECTURE

This model was represented in 2015 and considered famous deep learning-based object detector, consisting of three main parts [6]:

- Convolution Neural Network (CNN).
- Region Proposal Network (RPN).
- Class Detection Bounding Boxes prediction.

While the CNN generates the feature map from the input image, RPN predicates whether an object existed, or not then a bounding box for this object is created; in the end, there is a classifier (detects class of object) and regressor (creates a bounding box of the object).

© 2021 IJSRET

International Journal of Scientific Research & Engineering Trends
Volume 7, Issue 1, Jan-Feb-2021, ISSN (Online): 2395-566X

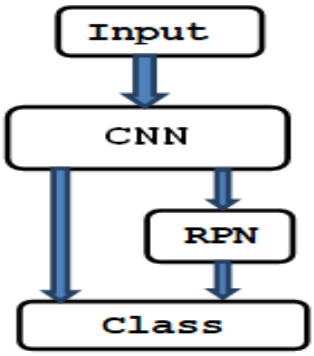

Fig 2.R-CNN Architecture.

CNN part in this model use basically the VGG16 pre-trained model [7], we can consider the RPN network is designed especially to generate bounding boxes, while the final network is a fully connected network.

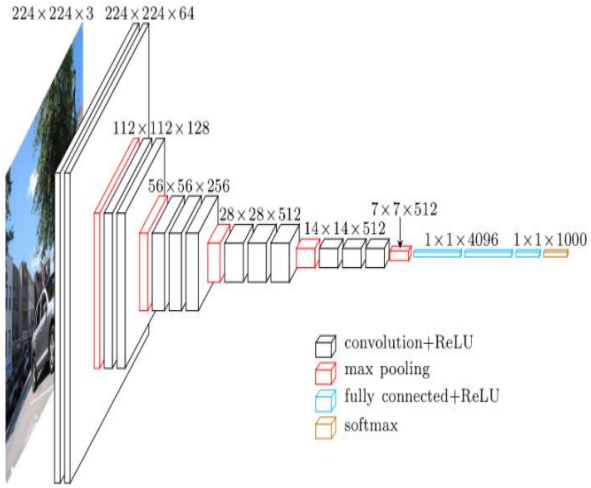

Fig 3.VGG16 Architecture.

Faster R-CNNmodel is 2 stages object detector (features map extracting and regions proposals generations), producing 71% mAP at 10FPS.

The loos function for an image in this model [8]:

$$\mathbf{L}(\{\mathbf{p_i}\},\{\mathbf{L_i}\}) = \frac{\mathbf{1}}{\mathbf{N_{cls}}}\sum_{\mathbf{i}}\mathbf{L_{cls}}(\mathbf{p_i},\mathbf{p_i^*}) + \boldsymbol{\lambda}\frac{\mathbf{1}}{\mathbf{N_{reg}}}\sum_{\mathbf{i}}\mathbf{p_i^*}.\mathbf{L_{reg}}(\mathbf{t_i},\mathbf{t_i^*})$$

Where:
i= index of an anchor in a mini batch,
p_i= predicted probability of anchor i being an object
p_i^*= ground truth label,is 1 if the anchor is positive,and is 0 if the anchor is negative
t_i= is a vector representing the coordinates of the predicted bounding box
t_i^*= ground truth box associated with a positive anchor
L_cls=classification loss ,log loss over two classes (object vs.not object)
L_cls = regression loss,function (smooth L1)
N_cls= mini batch size ,
N_reg= number of anchor locations,
λ= balancing parameter

III. SSD ARCHITECTURE

SSD single shot detector [10] is a single stage object detector, using only features map from the CNN block makes this model faster than 2-stages detectors, although SSD gained the fast performance; the model cannot detect small objects.

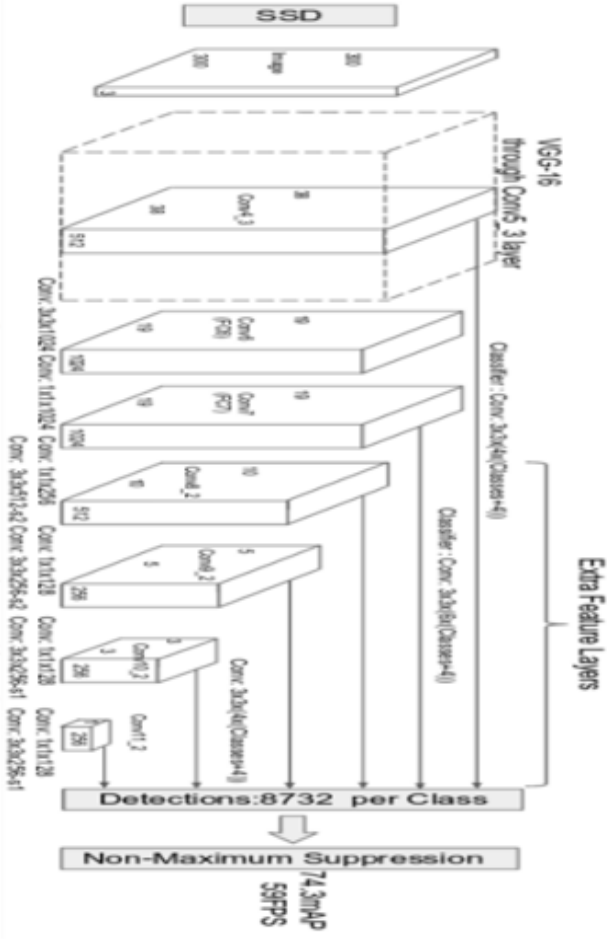

Fig 4.SSD Architecture.

This model is more efficient than Faster R-CNN due to feature maps are extracted through the convolutional network, not only at the end of the network.

© 2021 IJSRET

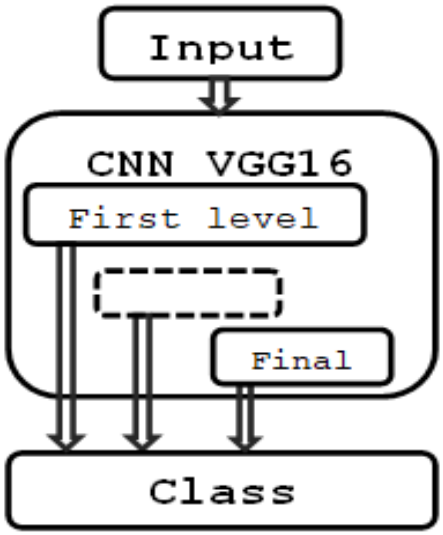

Fig 5.SSD Main Blocks.

New proposals came to exist to achieve more accuracy; using deconvolution and prediction modules helps the SSD to perform more accurately. But even by these final upgrades, the results didn't even close to YOLOv2 ones.SSD model offers a 74.3% mAP at 59 FPS.

The loose function [11]:

$$L(x,c,l,g) = \frac{1}{N}(L_{conf}(x,c) + \alpha L_{loc}(x,l,g))$$

Where,
L_conf (x,c):confidence loss is the loss in making a class prediction.
L_loc (x,l,g): localization loss is the mismatch between the ground truth box and the predicted boundary box.
N: is the number of positive match and α is the weight for the localization loss.

IV. YOLOV2 ARCHITECTURE

This model provide 78.6% mAP at 40FPS; the network can work for higher rate such 70 FPS; let's review the main techniques [12] used in this model which made that leap in speed and accuracy, making YOLOv2 is state-of-the-art real-time object detection model:

1.Batch Normalization: Normalize outputs of hidden layers making the learning process faster.
2.High-Resolution Classifier: Using the 448*448 classifier increases the accuracy of the model.
3.Use Anchor Boxes For Bounding Boxes: Build bounding boxes is implemented in C++ and smart technique depending on the object shape, which improves the speed and accuracy.
4. Darknet Deep Learning Neural Network: Fully convolutional neural network written in C++,making the model faster.
5. Direct Location Prediction Method: Making the network more stable.
6. No Fully Connected Layers.

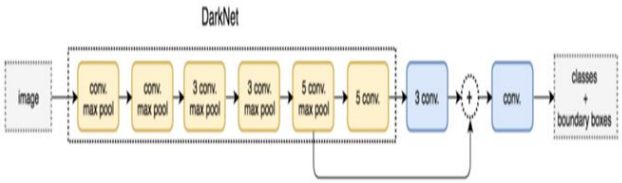

Fig 6.YOLOv2 Main Architecture.

The DarkNet is the backbone for the YOLOv2 model; it depends on an astonishing method of extracting features from the source image; as we go deeper in the net, the more convolutional layers so the number of layers is varying from one to five.

Table 1.DarkNet architecture removing last three layers.

Type	Filters	Size/Stride	Output
Convolutional	32	3 × 3	224 × 224
Maxpool		2 × 2/2	112 × 112
Convolutional	64	3 × 3	112 × 112
Maxpool		2 × 2/2	56 × 56
Convolutional	128	3 × 3	56 × 56
Convolutional	64	1 × 1	56 × 56
Convolutional	128	3 × 3	56 × 56
Maxpool		2 × 2/2	28 × 28
Convolutional	256	3 × 3	28 × 28
Convolutional	128	1 × 1	28 × 28
Convolutional	256	3 × 3	28 × 28
Maxpool		2 × 2/2	14 × 14
Convolutional	512	3 × 3	14 × 14
Convolutional	256	1 × 1	14 × 14
Convolutional	512	3 × 3	14 × 14
Convolutional	256	1 × 1	14 × 14
Convolutional	512	3 × 3	14 × 14
Maxpool		2 × 2/2	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	512	1 × 1	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	512	1 × 1	7 × 7
Convolutional	1024	3 × 3	7 × 7

The block between DarkNet and the classifier is called Neck [14]; its purpose, to extract different feature maps of different stages (features fusion) of the DarkNet, the thing which increases the accuracy of YOLO.

The loose function [15] for the YOLOv2 model is more complex than the previous functions for SSD and Faster RCNN;it includes classification, localization, and confidence losses functions.

Although YOLOv3 and v4[16] improve the performance in speed, the accuracy is decreased.

© 2021 IJSRET

V. APPROACH TO NEAR ACCURATE REAL-TIME OBJECT DETECTION

We will introduce an approach which upgrades the YOLOv2 model to be more accurate and even faster to work at frame rates like 100 FPS; there are three main points we can work on to adopt the model to be better:

1. Selection of Anchor Boxes:
In the YOLOv2, a traditional Euclidean distance function [17] is applied, which is used to find the nearest centroid to the ground-truth bounding box; it generates an error for large boxes. We can use k-means++ algorithm [18] to analyze size of bounding boxes and use the following metric [19]:

$$d(box, centroid) = 1 - IOU(box, centroid)$$

$$IOU = \frac{Area\ of\ Overlap}{Area\ of\ union}$$

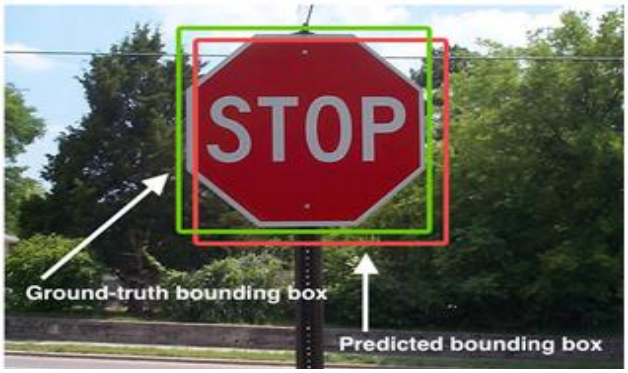

Fig 7.An Example of Predicted Bounding Box and Calculation of Metric.

This method will improve the position accuracy and increase mAP.

2. Improvement Of Loose Function:
We shall use the normalization batch technique in the whole network, not only at the start, which means after each convolutional block, we have a normalization process; that manage us to have dynamic bounding boxes to the objects as they move in the image.

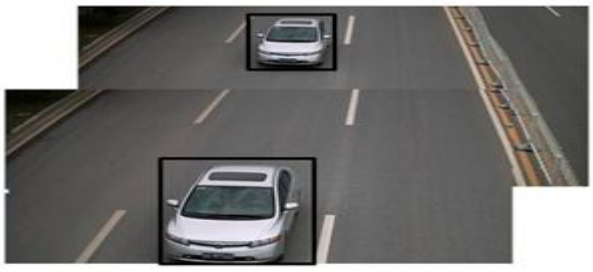

Fig 8.Comparison of the Same Object with Different Distances.

The loose calculation will be improved and increase the mAP by about 2%.

3. Network Design:
We can improve the design of YOLOv2 network by two steps:

3.1 Removing the repeated convolutional layers in each block; Instead of using 5 layers in the last blocks, we can use 3 in each block:
The main point here is developing a dynamic design based on YOLOv2; detecting large objects like humans, cars, and houses doesn't need that number of convolutional layers at the high block; it makes the model more complex only. We will have a faster model in the result; the thing allows us to do the next step.

3. 2 Upgrade features fusion:
In YOLOv2, there is only one line for features fusion; we can develop the design since we've managed to get free time-space from the first step, making the second line of features fusion extracting the data from the lower layer will enhance the model and improve the accuracy.
Figure 9: Modified network for YOLOv2 model.

VI. CONCLUSION

In this paper, we reviewed three models considered as state-of-the-art object detection, comparing the three models in architecture and lose function, explaining the techniques being used in YOLOv2, which make this model faster and more accurate than Faster-RCNN and SSD models.We presented many ways to develop the YOLOv2 model, trying to improve the accuracy and speed of processing.

Taking into consideration, developing the perfect real-time model cannot be achieved in the same hardware, we must think of upgrading the proceeding units, especially in the case of offline training and testing.

REFERENCES

[1] Deep Learning-Based Real-Time Multiple-ObjectDetection and Tracking from Aerial Imagery via aFlying Robot with GPU-Based Embedded Deviceshttps://www.mdpi.com/1424-8220/19/15/3371
[2] YOLOv2–Object Detectionhttps://www.geeksforgeeks.org/yolo-v2-object-detection/
[3] mAP (mean Average Precision) for Object Detectionhttps://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173.

© 2021 IJSRET

[4] Fast R-CNN https://openccess. com/ ontent_iccv_2015/papers/Girshick_Fast_R-CNN_ICCV_2015_paper.pdf
[5] Real-Time Brand Logo ps://link.springer.com/chapter/10.1007/978-3-319751931_14
[6] A deeper look at how Faster-RCNN workshttps://.com/a-deeper-look-at how-faster-rcnn-works 84081284e1cd#:~:text=At%20the%20conceptual%20level%2C%20Faster,a%20few%20last%2Ftop%20layers.
[7] VGG16–Convolutional Network for Classification andDetectionhttps://.io/en/popularnetworks/vgg16/
[8] Faster R-CNN: Towards Real-Time Object Detection withRegionProposalNetworkshttps://arxiv.org/pdf/1506.01497.pdf
[9] SmoothL1 Losshttps://mohitjainweb.files.wordpress.com/2018/03/smoothl1loss.pdf
[10] Variations of SSD—Understanding Deconvolutional Single-Shot Detectorshttps://medium.com/@amadeusw6/variations-of-ssd-understanding-deconvolutional-single-shot-detectors-c0afb8686d03
[11] SSD object detection: Single Shot MultiBox Detector for real-time processinghttps://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06#:~:text=Loss%20function,Negative%20matches%20can%20be%20ignored.
[12] How YOLOv2 works in detailhttps://medium.com/@y1017c121y/how-does-yolov2-work-daaaa967c5f7
[13] An overview of object detection: one-stage methods.https://www.jeremyjordan.me/object-detection-one-stage/
[14] DarkNet-19 convolutional neural networkhttps://www.mathworks.com/help/deeplearning/ref/darknet19.html
[15] Object Detection using YOLOv2 on Pascal VOC2012 losshttps://fairyonice.github.io/Part_4_Object_Detection_with_Yolo_using_VOC_2012_data_loss.html
[16] YOLO v4: Optimal Speed & Accuracy for object detectionhttps://towardsdatascience.com/yolo-v4-optimal-speed-accuracy-for-object-detection 79896ed47b50
[17] Euclidean Distan ps://www.sedirect.com/topics/mathematics/euclidean-distance
[18] K-means calculation anchor boxeshttps://www.prmersought.com/article/7534217961/
[19] Intersection over Union (IoU) for object detectionhttps://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

© 2021 IJSRET