You Only Look Once:
A comparative study for real-time object detection

Shrey Vaghela, Vyom Jain

Institute of Technology, Nirma University

Abstract

This paper is an attempt to compare and study state-of-the-art models used for real-time object detection. With the special focus on YOLO [3][4][5] models which have evolved over the span of 3 years, we will study, compare them with other algorithms like the SqueezeDet [2], FastYOLO [6] and VideoYOLO [7]. Understanding the differences and the similarities between these architectures will help us to have a deep understanding of the state of object-detection.

We will witness how the models have changed thus far and how other researchers influence others to create something new or to improve the existing one. We will also this through a timeline starting from 2015, with Faster R-CNN [9], going all the way to 2018, with the latest improvement YOLOv3 [3].

1. Introduction

With the rapid development in architectures for CNN from AlexNet, to VGG Net and then to ResNet, there is the great improvement in the application of computer vision. Different applications such as image classification, object detection, segmentation etc. can be implemented with higher efficiency. Object detection being one of many applications is used to locate and label different multiple objects in the images. It is not only restricted to images but can also be applied to videos and for real-time object detection.

Most of the application of detection demands it to be real time. Current real-time state-of-the-art technologies such as YOLO and SSD are based on R-CNN, a milestone for object detection in images [1]. The methodology of detecting objects differs in YOLO and its previous architectures. For real-time detection, the algorithm not only has to be faster but also efficient enough in order to fulfil the requirements for applications such as self-driving cars, smart assistive devices or robotic systems. The trade-off between the speed and accuracy should be taken care according to the application and platform [8].

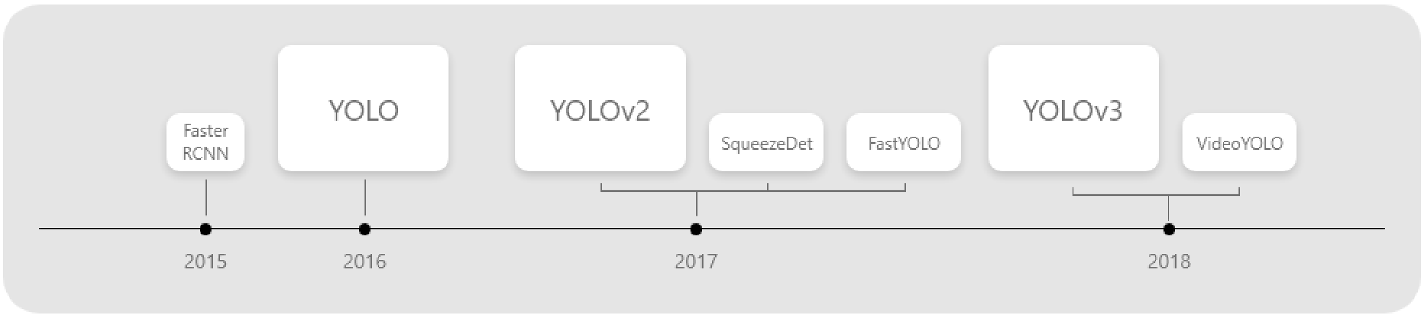
We can always trade-off between accuracy and speed by just changing the model size without even retraining it. We can achieve 45 fps with object detection on real-world entities, also one can attain 155 fps but is less accurate. Also, many applications such as embedded systems require real-time object detection even with low computation power and memory [6]. Thus, an algorithm with fewer parameters needs to be developed.

2. Various attempts

Many methods including R-CNN, Fast R-CNN etc. exists which uses CNN but fails to perform well due to their mechanism of cropping the part of an image with or without feature extractor [8]. Thus, it became important to find a new approach for real-time object detection.

2.1 YOLO

YOLO is short for You Only Look Once. As the name suggests it just sees or more technically runs the whole CNN only once. YOLO then after passing the whole image through network predicts the boundary box and then calculates the probability for the class label. It does detection by regression [4].



Unlike R-CNN, Faster R-CNN where region proposal networks are first used to propose the bounding boxes followed by running the classifier on top of the proposed region in an image, YOLO represents object detection as a regression problem. All of the processes take place in a single pass of the convolutional neural network predicting multiple bounding boxes and class probabilities simultaneously. In YOLO, the network sees an image as a whole. As it sees the larger context, the number of background errors reduce.

The algorithm divides the image into grids and each grid predicts bounding boxes and confidence of that box whether it contains object or not. The confidence is calculated by Pr (Object) * IOU. Instead of four coordinates of the bounding box, it predicts the centre, width of the bounding box, the height of the bounding box and a confidence score [4]. Along with these it also predicts Pr (Class Object) as conditional class probabilities.

However, it is fast but the disadvantage because of this mechanism is that there is an increase in the error for localization. It struggles to identify small objects in a group. Also, images with new aspect ratio make it difficult to detect the same objects [4].

2.2 YOLO 9000

YOLO was already very fast and accurate but had some shortcomings. These shortcomings and instabilities convinced them to write a paper which was a monumental improvement over YOLO.

- YOLO makes a significant number of localization errors compared to faster R-CNN [9]
- Lower recall compared to region proposal-based algorithms [10]
- Anchor boxes dimensions are hand-picked
- Model instability, problem with anchor boxes. Comes from predicting (x, y) locations from the box.
- Fixed to only single input dimension of 448x448.
- Uses a custom model for classification based on GoogLeNet [11], but has slightly worse accuracy than VGG-16[12].

The architecture was improved to accommodate 9000 classes and was able to detect even those objects which didn't have labelled data. Moreover, dropout normalization was completely replaced by batch normalization which improved the accuracy by 2%. The model was trained for high-resolution images for 10 epochs which improved it further by 4%. Prediction of bounding boxes is redefined by predicting offsets by convolution layers instead of handpicked ones. Uses an odd number of locations in feature maps so that bigger objects which tend to occupy the centre are classified for only one centre pixel, not the surrounding four.

Anchor box sizes are no longer a hyperparameter but are learned using k-means clustering. Takes inspiration from ResNet [13] by adding passthrough features of 26x26 along with the usual 13x13 features. Improved accuracy by 1%. The model has been made robust to different image resolutions by changing the resolution in every few epochs, going from 320x320 to 608x608. Uses DarkNet-19 instead of the one used in YOLO [4].

To make the model more robust, the detection and classification datasets were mixed and losses were handled differently for them. The data used is multi labelled which doesn’t assume mutual exclusion. This is done by not using SoftMax. Labelling is done by using WordNet, which is a graph. The model uses only a hierarchical tree from the concepts of ImageNet [15].

2.3 YOLOv3

YOLOv3 is regarded as an upgrade to the previous version YOLO 9000. Joseph Redmon and Ali Farhadi themselves call it not a research paper but a TECH REPORT. This time they take inspiration from the state of the art and common knowledge to improve the model.

Logistic regression is used to predict the objectness score of each bounding box. The value is 1 for the block which has the maximum overlap with the ground truth. This is done without using SoftMax to incorporate multilabel data. Binary cross-entropy is used during training. This helps in datasets like the Open Images dataset which has overlapping labels like person and woman.

Boxes are predicted at 3 different scales. Several convolutional layers are placed upon the basic feature extractor to get a 3-D tensor encoding bounding box, objectness and class prediction. To get more semantic details the paper highlights the use of features from 2 layers before and upscaling them by 2x. Also, features from other previous layers are concatenated and then convoluted to get finer-grained features from the earlier feature maps.

A new feature extractor is used. Instead of DarkNet-19 used in YOLO 9000[5], they use a hybrid of DarkNet-19 and ResNet. This network has 53 convolutional layers hence the name DarkNet-53. This is more powerful than DarkNet-19 and more efficient than ResNet-101 or ResNet-152. Network structure better utilizes the GPU. It is better for detecting smaller objects. The earlier YOLOs struggled with small objects. The trend is reversed in this one, YOLOv3 seems to struggle with medium to large size objects.

2.4 Fast YOLO

YOLOv2 being state of the art deep neural network achieves great performance in terms of speed and accuracy with powerful GPUs. But it still remains challenging for YOLOv2 to give the performance in a video for embedded systems with limited computational power and memory. Motion adaptive method has been introduced to Fast YOLO to reduce computational power [6]. It includes 2 steps:

1) optimization of YOLOv2 and
2) motion-adaptive inference.

For each video frame, an image stack is passed to 1x1 convolutional layer for calculating motion probability. The paper discusses the evolutionary deep intelligence framework where probabilistic DNA of an ancestor network and environmental factors are taken into consideration to produce new deep neural network [6]. The environmental factors are used such that the number of parameters is reduced and YOLOv2 is optimized. All the frames in the video do not require the computation. Thus, if the frame is unique with

Paper	***Architecture***	***Year***	***Distinguishing feature***
YOLO	Based on GoogLeNet	2016	First attempt towards Fast real-time object detection
YOLO 9000	DarkNet-19	2017	Incorporated 9000 classes
YOLO v3	DarkNet-53	2018	Better for smaller images and more accurate
FastYOLO	1x1 conv layer for motion probability map.	2017	Better for embedded systems
SqueezeDet	ConvDet is a layer with W_{filter}*H_{filter} convolution and output size of k*(5+C)	2017	Conv layers are used to compute bounding boxes and class probabilities. Small, low powered
VideoYOLO	8 Conv3 layers, 5 MaxPool layers, 2 fc layers and 1 SoftMax layer	2018	Sampling from distinct distributed frames gives larger context of the video

Table 1: A comparison of all the models discussed in this paper. At a glance we can see the advancements made in the field of object detection over the years

reference to the reference frame then with the help of the motion probability map we compute updated class probabilities and that unique frame now becomes our reference frame for future video frames.

Using this approach one can increase the speed of object detection by approximately 3.3 times and reduce the deep inference by 38.13% [6].

2.5 SqueezeDet

The network named SqueezeDet is inspired by YOLO for its single stage detection pipeline. At the initial stage of the pipeline, low resolution and high dimensional feature map is extracted and fed to ConvDet [2]. ConvDet is a convolutional layer, which computes bounding boxes and its confidence score along with conditional class probability:

$$\max \Pr(\text{class} \mid \text{Object}) * \Pr(\text{Object}) * \text{IOU}$$

We keep top N bounding boxes and use Non-Maximum Sequence to filter and get final detection. With the help of ConvDet, one can propose thousands of region proposals with lesser parameters in comparison to YOLO [4]. ConvDet works by using a sliding window approach. Reference boxes known as anchors with pre-selected shapes are used to compute four coordinates of the predicted bounding boxes. The ConvDet is similar to the last layer of Region proposal network in Faster R-CNN. Both of them majorly defers in implementing the classifiers where ConvDet implements CNN and Faster R-CNN implements Fully Connected Layer where RPN is only responsible for generating box proposals.

2.6 VideoYOLO

Being inspired by the paper of Joseph Redmon on YOLO where object detection and classification takes place in a single forward pass on full images, this paper discusses the implementation of action recognition by learning temporal

characteristics in a single pass. Video YOLO captures overall temporal information spread across the whole video [7]. This capturing of subsets of frames across the video can be done by 1) even sampling or 2) random sampling.

$$S_i = 1 + \lfloor N/T \rfloor * i, \qquad (1)$$

$$S_i = \text{random}(\lfloor N/T \rfloor) + \lfloor N/T \rfloor * i, \qquad (2)$$

Here, N = length of the video, T = number of frames to be sampled. Random sampling is done by dividing the video into T intervals uniformly. Then, from each interval, one frame is selected randomly with the help of random(⌊N/T⌋). These two sampling methods are used to make the proxy video which is then fed to 3D CNN for learning the appearance and global temporal information.

3D-CNN used has 11 layers including 8 convolutional layers followed by 2 FCL and a SoftMax layer. Even though different proxy videos might have different lengths, but keeping the same number of parameters makes it easier to compare different networks. 3D-CNN can learn even with fewer frames. Also, the efficiency increases as the number of frames during sampling increases [7].

3. Conclusion

Object detection is one of the applications of Machine Learning which is being used extensively in the industry. With changing times and datasets, we need the models to be more robust, accurate and faster at the same time. Advances in this field have been rapid as we have seen in this paper. Right after the publication of Faster R-CNN [9] and YOLO [4], many papers have been published which are either improvement of these or are a new approach for object detection. This paper is an attempt to study and compare all the major papers which were published in this field. This has helped us to understand object detection in a wholesome sense which is pivotal before applying any technique to solve a real-world problem.

References

[1] Du, J., 2018, April. Understanding of Object Detection Based on CNN Family and YOLO. In Journal of Physics: Conference Series (Vol. 1004, No. 1, p. 012029). IOP Publishing.

[2] Wu, B., Iandola, F.N., Jin, P.H. and Keutzer, K., 2017, July. SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving. In CVPR Workshops (pp. 446-454).

[3] Redmon, J. and Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[4] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[5] Redmon, J. and Farhadi, A., 2017. YOLO9000: better, faster, stronger. arXiv preprint.

[6] Shafiee, M.J., Chywl, B., Li, F. and Wong, A., 2017. Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video. arXiv preprint arXiv:1709.05943.

[7] Jing, L., Yang, X. and Tian, Y., 2018. Video you only look once: Overall temporal convolutions for action recognition. Journal of Visual Communication and Image Representation, 52, pp.58-65.

[8] Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S. and Murphy, K., 2017, July. Speed/accuracy trade-offs for modern convolutional object detectors. In IEEE CVPR (Vol. 4).

[9] Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster r-CNN: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

[10] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-CNN: Towards real-time object detection with region proposal networks. arXiv preprint arXiv:1506.01497, 2015.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014

[12] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385, 2015

[14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft coco: Common objects in context. In European Conference on Computer Vision, pages 740–755. Springer, 2014

[15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), 2015