Evaluation and Evolution of Object Detection Techniques YOLO and R-CNN

K G Shreyas Dixit, Mahima Girish Chadaga, Sinchana S Savalgimath, G Ragavendra Rakshith, Naveen Kumar M R

Abstract: ***Object detection has boomed in areas like image processing in accordance with the unparalleled development of CNN (Convolutional Neural Networks) over the last decade. The CNN family which includes R-CNN has advanced to much faster versions like Fast-RCNN which have mean average precision(Map) of up to 76.4 but their frames per second(fps) still remain between 5 to 18 and that is comparatively moderate to problem-solving time. Therefore, there is an urgent need to increase speed in the advancements of object detection. In accordance with the broad initiation of CNN and its features, this paper discusses YOLO (You only look once), a strong representative of CNN which comes up with an entirely different method of interpreting the task of detecting the objects. YOLO has attained fast speeds with fps of 155 and map of about 78.6, thereby surpassing the performances of other CNN versions appreciably. Furthermore, in comparison with the latest advancements, YOLOv2 attains an outstanding trade-off between accuracy and speed and also as a detector possessing powerful generalization capabilities of representing an entire image.***

Keywords: CNN, R-CNN, Fast R-CNN, Faster R-CNN, YOLO, Image processing, Object detection

I. INTRODUCTION

Extracting objects from natural images can be considered as elementary and reproving issues in the field of computer vision along with image processing. It has a prominent part in vision applications which include classification, recognition of objects, etc. Nevertheless, experiments in this field have shown that it persists as a pending and demanding issue. This is because images of natural environments indicate real-world adaptations which are distinguished by a vast scope of shapes, colors along with textures. The goal of object detection is to locate an object in a particular photograph accurately then categorize objects accordingly. This task can be easily performed by the human eye but a computer processes image in a distinct way. It takes into that two dimensions also plays a crucial part in the object's size, space, orientation, attitude and place in the image.

Object detection helps in vehicle detection, surveillance, pose approximation, etc. In detection algorithms, around each object of importance, one bounding box is drawn to locate that object in that image. Similarly, multiple bounding boxes could be drawn to locate multiple objects in an image which might not be known ahead of time.

In recent days, CNN is considered as a powerful machine learning tool which can be used in a wide variety of applications like face detection, video recognition, voice recognition, etc. It can bring an input image, allocate weights and biases to different items in the image, and can distinguish between them.

Recent techniques like R-CNN, Fast R-CNN make use of region proposal techniques to create bounding boxes in a photograph [1]. After classification, another level of processing is carried out to rectify bounding boxes, remove similar detections and reassign the scores for each box considering all the other objects in the photograph [2] [3].

The previous object detection techniques made use of classifiers to perform detection. Instead of considering the task of detecting an object as that of a classification one, YOLO considers it a regression one in order to dimensionally separate the bounding boxes and associate their class probabilities. A single network splits the image into many portions, generates bounding boxes and class probabilities for every portion being an object. It is capable of predicting bounding boxes along with their class probabilities from a photograph in a single analysis. Each bounding box is given weights based on the probabilities predicted [4]. This network can be optimized from end to end based on the performance of detection because there is a single network involved. This sort of construction makes the network really fast [5].

II. CNN FAMILY

CNN - The convolutional neural network, it is a type of feed-forward network that can extract topological features from an input image. First, the network removes the features from a raw image and then classifies the extracted features using a classifier. CNN's are steady networks. That is, CNN's are unvarying to elementary geometric transformations such as scaling, translations, squeezing and rotation and deformations. CNN's most prominent characteristic feature is its ability to reduce the parameter count in ANN [6].

The architecture of convolutional neural network prominently consists of three layers namely, convolution layer, pooling layer, and the fully connected layer. A feature

Revised Manuscript Received on July 10, 2019

K G Shreyas Dixit, Department of Information Science and Engineering, Vidyavardhaka College of Engineering, Mysuru, Karnataka, India

Mahima Girish Chadaga, Department of Information Science and Engineering, Vidyavardhaka College of Engineering, Mysuru, , Karnataka India

Sinchana S Savalgimath, Department of Information Science and Engineering, Vidyavardhaka College of Engineering, Mysuru, Karnataka, India

G Ragavendra Rakshith, Department of Information Science and Engineering, Vidyavardhaka College of Engineering, Mysuru, Karnataka, India (Email: Rrakshithgr@outlook.com)

Naveen Kumar M R, Department of Information Science and Engineering, Vidyavardhaka College of Engineering, Mysuru, Karnataka, India

vector of fixed-length from the feature map is extracted for each and every object proposal by the pooling layer [7]. All possible feature vector is then fed to series of fully connected layers which in due course branches into sibling output layers, one amongst this layer generates softmax probability estimation over K classes of the object along with a catch-all background, class and the second layer, for each of the K object classes outputs four real-valued numbers. The convolutional neural network architecture comprises prominently of three parts: the convolution layer, the pooling layer, and the fully connected layer. The pooling layer [7] extracts a fixed-length function matrix from the function chart for each and every item proposition. All function vectors are then supplied to the sequence of fully connected layers that branch into sibling input layers in due course, one of these layers produces softmax likelihood estimates over object groups along with a catch-all background, class and second layer for each of the object groups produces four real-valued numbers.

Convolution Layer: CNN's key portion is the convolution layer, consisting of the local links and the weights of the mutual features. This layer's goal is to understand input function representations. The layer of convolution consists of several function maps. Each neuron of the identical function map is used to remove the local features of distinct locations in the preceding phase, while this function map is used to remove the local features of the same locale for the following cells. To get a fresh feature, a taught kernel is first converted, followed by input feature maps, and the result is then transferred to a nonlinear activation function. Similarly, divergent feature maps can be obtained by the application of different kernels [8].

The pooling layer: Pooling layer is generally placed in the middle of two convolution layers. Pooling layer has got the second feature extraction effect, this reduces feature maps dimensions and increases the feature extraction robustness. The feature maps size in pooling layer is determined in accordance with the moving step of the kernel. Max and Average pooling are the two pooling operations [8]. By stacking a pile of convolution and pooling layers the higher level features of an input can be extricated [9].

Fully connected layer: In a Convolutional neural network classifier, one or more fully connected layers may be present. Every single neuron from a layer is linked in the instant next layer to every other single neuron current. The last fully connected layer precedes the output layer and this layer does not have any spatial information within it [8].

R-CNN - R-CNN object detection basically comprises of three modules. The first module is related to the production of region proposals that are category independent. The set of candidate detections that are available to the detector are described by these proposals. The second module, which is an enormous convolutional neural network, is concerned with extrication of a feature vector of fixed size from all possible region. The last module is composed of a pile of linear SVMs that are class specific [10].

Fast R-CNN - Fast R-CNN is a network in which an entire image together with a set of object proposals is taken in as input. The network produces a conv feature map which processes the complete image with the help of a variety of convolutional layers. The max n system is made up of two modules. The fully convolutional network is the foremost module and this network proposes the regions. Fast R-CNN detector which makes use of the regions proposed is the last module [11].

The main difference between them when it comes to R-CNN and Fast-CNN is that in R-CNN the regional detection proposals are inputted at the pixel level, whereas in Fast R-CNN the regional proposals are input at the feature map level.

Faster R-CNN - Faster R-CNN is a network in which the same convolution network carries out the task of object detection and generation of region proposal [12]. Object detection is much faster with this kind of design. The solution for the regional proposal problem is given by Faster R-CNN with the addition of RPN and this is Faster R-CNN's main contribution. The input to the ROI pooling is not the original image instead it is the regional proposal on the final feature image. When compared with all the previous CNN models, Faster R-CNN's computation is very less. This is due to the fact that the feature image's resolution is lesser than that of the original image [13].

III. COMPARISON BETWEEN CNN FAMILY

The advancement of object detection technology follows the R-CNN - SppNET - Fast R-CNN -Faster R-CNN order.

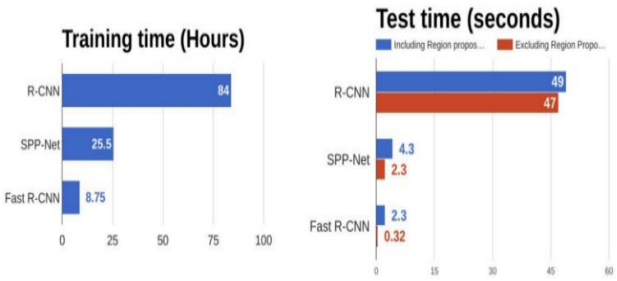

Figure 1: Training and Testing time comparison for CNN [14]

From the graph in Figure 1, it is evident that the training and testing session is remarkably faster in Fast R-CNN when compared with R-CNN [14]. There is a notable slowdown in the algorithm which enframes region proposals in comparison with the one not using them when the Fast R-CNN's performance during testing is considered [15]. Thus in Fast R-CNN algorithm region proposals act as hold-ups influencing its performance.

Table 1: Speed comparison for CNN [16]

	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

The major handout of the Faster R-CNN is that it designs an RPN network that extricates candidate areas instead of wasting time on the selective search, which remarkably accelerates the detection [16]. Table 1 shows the speed comparison between R-CNN, Fast R-CNN, and Faster R-CNN. The Faster R-CNN's main handout is that it builds an RPN network that extricates applicant regions rather than spending time on the selective search, which significantly accelerates detection [16]. Table 1 demonstrates the speed comparison between R-CNN, Fast R-CNN, and Faster R-CNN.

IV. YOLO

Another approach to object detection is YOLO (You only look once). What the objects are in an image and where they are present can be predicted by looking only once at the image. Instead of considering the task of detecting an object as that of a classification one, YOLO considers it a regression one in order to dimensionally separate the bounding boxes and associate their class probabilities [13]. A single network splits the image into many portions, generates bounding boxes and class probabilities for every portion being an object. It is capable of predicting bounding boxes along with their class probabilities from a picture in a single analysis. Similarly, a single convolutional network is able to multiple bounding boxes and their class probabilities. Each bounding box is given weights based on the probabilities predicted. This network can be optimized from end to end based on the performance of detection because there is a single network involved [4] [17].

YOLOv1: The input image is divided into an SxS grid. If the center of a particular object falls in the center of a grid cell, then that grid cell has the responsibility of detecting that object [18]. The grid cell has the job of predicting bounding boxes(B), confidence scores and class probabilities(C) for each grid [6]. These predictions can be formulated as $S \times S \times (B * 5 + C)$, which is a tensor. The confidence scores indicate the confidence level and accuracy level of the model in predicting that an object is contained in the box. The confidence score can be defined as Pr(Object) ∗IOUtruth pred where IOU stands for intersection over the union. The confidence score is zero if there isn't any object that exists in that cell otherwise it should be the value of IOU between ground truth and predicted box [4].

During testing, conditional class probabilities and confidence predictions for individual boxes are multiplied in order to get class specific scores for every box.

Pr(Classi|Object) ∗ Pr(Object) ∗ IOUtruth pred = Pr(Classi) ∗ IOUtruth pred [3]

The network architecture of YOLO version 1 has taken inspiration from a network model named GoogleNet which was designed for image classification. This network consists of 24 convolutional layers and 2 fully connected layers. The alternating 1x1 convolutional layers and reduction layers help in reducing feature space from the preceding layers [4].

YOLOv2: Yolo v2 is a new method which is used to harness a huge quantity of classification data which is present in order to develop the scope of existing techniques. This technique allows the combining of distinct datasets together using an ordered perspective of object classification. A joint training algorithm enables the training of object detectors on detection as well as classification data. This technique allows the images to accurately comprehend the localization of objects and it makes use of these classified images to escalate its robustness and vocabulary [19].

In YOLOv2 firstly the classification network is fine-tuned on ImageNet at a full resolution of 448x448 for around 10 epochs. This helps the network in adjusting its filters so that it can work better on inputs with high resolution. The network is then fine-tuned. This classification network of high resolution gives a 4% increase in mAP.

In this version of YOLO, the prediction of bounding boxes is done by anchor boxes instead of fully connected layers. Firstly, a pooling layer is removed so as to escalate the network's output resolution. .In Order to operate the network on 448x448 instead of 416 input images, the network is shrunk [19].

YOLOv3: YOLOv3 makes use of logistic regression to predict the objectness score for the bounding boxes. The objectness score should be 1 if a ground truth is overlapped by a bounding box prior greater than another bounding box. The prediction can be disregarded if a bounding box prior overlaps the ground truth by a certain threshold but isn't the best. Boxes in YOLOv3 are predicted in three different scales. Then the features are extracted from each scale in a technique similar to that of feature pyramid network. Feature maps belonging to the previous two layers are unsampled twice. The feature map which already exists within the network is concatenated with unsampled features. This technique enables the gain of meaningful semantic and fine-grained information from unsampled features and previous feature map respectively. In YOLOv3, additional convolutional layers are included that process the combined feature map and to predict a twice sized tensor [20].

V. SSD

A convolutional network is once run by the SSD on an input image and then calculates a feature map. On the feature map, it runs a 3x3 sized convolutional kernel in order to predict the bounding boxes along with their class probabilities. SSD makes use of anchor boxes similar to Faster-RCNN at varied aspect ratios. Instead of learning the box, it learns the offset. SSD is capable of detecting objects at various scales because of every convolutional layer processes at a varied scale [16].

VI. COMPARISON BETWEEN CNN AND YOLO

Framework

Faster R-CNN and YOLO consider their core as CNN. Their main objectives are to find an improved way to divide the proposals using CNN. Though they have similar objectives, their frameworks are different from each other. RCNN at first keeps the whole image and divides the proposals later whereas YOLO divides the image before it uses CNN to process the image. YOLO does not make use of sliding windows or regional proposals but instead, the

input image is divided into grids. YOLO gave up on anchor boxes but YOLOv2 reused the anchor boxes. Faster-RCNN uses 9 anchor boxes but YOLOv2 uses just 5 anchor boxes [13].

Performance

Table 2: Speed and performance comparison [21]

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

Table 2 compares the speed and performance of fast detectors. For the most popular PASCAL VOC dataset, amongst the Yolo models, Fast YOLO proves to be the fastest [21]. But YOLO is more precise when compared to the faster version of YOLO.

YOLO is compared with other detectors on their GPU implementation. The relative maps and speeds of the detectors are compared to analyze performance and accuracy tradeoffs. For PASCAL VOC dataset, Fast YOLO turns out to be the fastest detector. With 52.7% map, it is two times as precise as any other fast detector. YOLO increases its map up to 66.4% whilst retaining its real-time performance [4].

Figure 2: speed vs accuracy trade-off [16]

Figure 2 shows that Faster-RCNN is the best choice of the algorithm if accuracy is needed and if accuracy is not much of a concern then super-fast YOLO can be chosen. SSD is recommended for computations running on Nvidia Jetsons [16] [22].

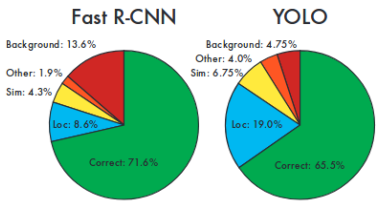

Figure 3: Error analysis for Fast R-CNN vs YOLO [4]

Figure 3 shows the proportion of background and localization errors across 20 classes. YOLO struggles while localizing objects accurately. In YOLO's errors, the majority of the errors are localization errors. Fast R-CNN makes more background errors and fewer localization errors [4].

Table 3: YOLOv2 on VOC 2007 [21]

Detection Frameworks	Train	mAP	FPS
Fast R-CNN [5]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[15]	2007+2012	73.2	7
Faster R-CNN ResNet[6]	2007+2012	76.4	5
YOLO [14]	2007+2012	63.4	45
SSD300 [11]	2007+2012	74.3	46
SSD500 [11]	2007+2012	76.8	19
YOLOv2 288 × 288	2007+2012	69.0	91
YOLOv2 352 × 352	2007+2012	73.7	81
YOLOv2 416 × 416	2007+2012	76.8	67
YOLOv2 480 × 480	2007+2012	77.8	59
YOLOv2 544 × 544	2007+2012	**78.6**	40

Table 3 is the comparison of YOLOv2 with other detection frameworks for PASCAL VOC 2007 [21]. The second version of YOLO is more accurate and fast when compared to the previous detection systems. YOLOv2 is an avant-garde detector achieving a map of 78.6% at high resolution on PASCAL VOC 2007 dataset. It also runs at various resolutions to achieve an easy trade-off between accuracy and speed [19].

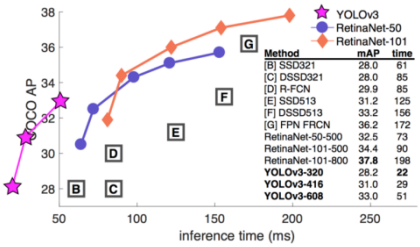

Figure 4: Performance of YOLOv3 with COCO [5]

VII. YOLO COMPARISON & ANALYTICAL RESULTS

YOLO VS YOLOv2

Table 4 summarizes the developments in YOLOv2. Several changes in design led to a remarkable increase in mAP [4].

Batch normalization: By including batch normalization on all of the convolutional layers, the map can be increased by about 2%.

High-Resolution Classifier: Initially, YOLOv2 trains the classifier with 224 x 224 images and later trains the classifier with 448 x 448 images by making use of fewer epochs. This improves the map by 4%.

Convolutional with Anchor Boxes: Using anchor boxes, there is a slight dip in accuracy levels. YOLOv2 with anchor boxes predicts around 1000 boxes per image whereas YOLO predicts only 98 boxes per image. Without anchor boxes, YOLO gets a recall of 81% with 69.5% map but with anchor boxes, YOLOv2 gets a recall of 88% with 69.2% map [23].

Fine-grained features: YOLOv2 adds a pass-through layer which brings features from the previous layers at a resolution of 26 x 26. This layer concatenates lower and higher resolution features. This is done by placing the adjoining features on different channels. This increases performance by 1%.

Dimension Clusters: In Yolov2, k-means clustering is performed on a set of bounding boxes used for training to instantly identify right priors instead of choosing them by hand.

Direct location prediction: Since the location prediction

is constrained, it makes it simpler for the network to learn the parameterizations which make the network sturdy. The direct prediction of the location of the center of a bounding box along with the use of dimension clusters improves the performance of YOLOv2 by 5%.

Multi-Scale Training*:* Rather than fixing the size of the input image, the network is changed after a few iterations. After a certain number of iterations, the network just chooses another different dimension for the images with no particular rule. The same network will be able to predict the detections for different resolutions of images. YOLOv2 presents an easy tradeoff between speed and accuracy since it runs faster for small sized images. Figure 5 shows the speed and accuracy of different object detection algorithms on VOC 2007 [19].

Table 4: Improvements from YOLO to YOLOv2 [19]

	YOLO							YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓
anchor boxes?				✓	✓			
new network?					✓	✓	✓	✓
dimension priors?						✓	✓	✓
location prediction?						✓	✓	✓
passthrough?							✓	✓
multi-scale?								✓
hi-res detector?								✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

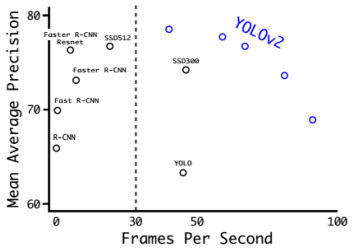

Figure 5: Accuracy and speed on VOC 2007 [19]

Figure 5 shows the accuracy and speed of various object detection algorithms in VOC 2007 [19].

YOLOv2 vs YOLOv3

In YOLOv3, a new network is used to perform feature extraction. It uses a combination of approaches ie Darknet-19 and a residual network. It makes use of successive 3 x 3 and 1 x 1 convolutional layer. It is called Darknet-53 because it has 53 convolutional layers [20].

Darknet-53 obtains high floating point operations per second. This network structure utilizes GPU much better and improves its evaluation efficiency and thus makes it faster [24]. YOLOv3 works remarkably quicker than the other detection systems with the equivalent performance [20].

Table 5: Darknet-53 [20]

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

For COCO dataset, the average mean AP metric for YOLOv3 comes up to the deviants of SSD but it is three times faster [24]. YOLOv3 is much stronger when compared to the old detection metric map at IOU=0.5. YOLOv3 is an extremely strong detector which outclasses others in producing satisfactory boxes for objects. For multi-scale predictions, YOLOv3 performs relatively high [20]. Table 6 summarizes the performance of YOLOv3.

Table 6: Comparison of YOLOv3 with other state-of-the-art models on the AP50 metric [20]

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 x 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

VIII. CONCLUSION

This paper is all about the present techniques in object detection, considering the CNN and You Only Look Once (YOLO). When YOLO is compared with CNN's, You Only Look Once(YOLO) can be used for many futuristic applications. YOLO is a coalesced detection model for objects. Building a YOLO model is easy and training the full images is effortless and straightforward. Contrasting the approaches of the classifier, the loss function is the heart of YOLO on which it is trained, detection performance is analogous to lose function and training of the whole model can be done together. In terms of general-purpose object detectors, the fastest YOLO version is fast YOLO. YOLOv2 provides the nonpareil quid pro quo between live speed and accuracy for detection of objects than the current systems across a wide variety of detection datasets. YOLOv3 using logistic regression predicts the boxes at 3 different levels. YOLOv3 performs very well in the fast detector category when speed is important. Moreover, applications that demand speed, robust object detection can depend on YOLO because YOLO generalizes representation of object other than models. These prominent points make YOLO a strongly recommended and widely spoken detection system. Other than the algorithm's structure, the most anticipated ultimatum for machine learning is the range of the set of

data. The attainability of acceptable data for training could be a crucial part in the process of learning is to attain an idea results.

IX. ACKNOWLEDGEMENTS

The manuscript is prepared by taking assistance from Accendere Knowledge Management Services Pvt. Ltd, we are thankful to them. We also express our gratitude to our teachers and mentors for guiding us throughout the work.

REFERENCES

1. "YOLO3: A Huge Improvement – mc.ai," 2018. [Online]. Available: https://mc.ai/yolo3-a-huge-improvement/. [Accessed: 10-January-2019].
2. S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1–6.
3. Ankit Sachan, "Zero to Hero: Guide to Object Detection using Deep Learning: Faster R-CNN,YOLO,SSD," CV-Tricks.com, 2018. [Online]. Available: https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/amp/. [Accessed: 27-January-2019].
4. Dhruv Parthasarathy, "A Brief History of CNNs in Image Segmentation: From R-CNN to Mask R-CNN," 2017. [Online]. Available: https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4. [Accessed: 03-February-2019].
5. J. Du, "Understanding of Object Detection Based on CNN Family and YOLO," J. Phys. Conf. Ser., pp. 1–8, 2018.
6. D. Foley and R. O'reilly, "An Evaluation of Convolutional Neural Network Models for Object Detection in Images on Low-End Devices," in AICS, 2018, vol. 2259, pp. 1–12.
7. G. Guo, H. Wang, Y. Yan, J. Zheng, and B. Li, "A Fast Face Detection Method via Convolutional Neural Network," Neurocomputing, pp. 1–23, 2018.
8. S. Guo, S. Chen, and Y. Li, "Face recognition based on convolutional neural network and support vector machine," in 2016 IEEE International Conference on Information and Automation (ICIA), 2016, pp. 1787–1792.
9. T. Guo, J. Dong, H. Li, and Y. Gao, "Simple convolutional neural network on image classification," in 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)(, 2017, pp. 721–724.
10. H.-J. Jeong, K.-S. Park, and Y.-G. Ha, "Image Preprocessing for Efficient Training of YOLO Deep Learning Networks," in 2018 IEEE International Conference on Big Data and Smart Computing (BigComp), 2018, pp. 635–637.
11. Jonathan Hui, "Object detection: speed and accuracy comparison (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and…," medium.com, 2018. [Online]. Available: https://medium.com/@jonathan_hui/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359. [Accessed: 10-February-2019].
12. H. Khalajzadeh, M. Mansouri, and M. Teshnehlab, "Face Recognition Using Convolutional Neural Network and Simple Logistic Classifier," 2014, pp. 197–207.
13. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," pp. 1–10, 2016.
14. J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," pp. 1–6, 2018.
15. J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6517–6525.
16. P. Ren, W. Fang, and S. Djahel, "A novel YOLO-Based real-time people counting approach," in 2017 International Smart Cities Conference (ISC2), 2017, pp. 1–2.
17. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in Advances in Neural Information Processing Systems 28 (NIPS 2015), 2015, pp. 91–99.
18. Rohith Gandhi, "R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms," 2018. [Online]. Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e. [Accessed: 15-March-2019].
19. Sik-Ho Tsang, "Review: R-CNN (Object Detection)," Coinmonks – Medium, 2018. [Online]. Available: https://medium.com/coinmonks/review-r-cnn-object-detection-b476aba290d1. [Accessed: 28-March-2019].
20. Sik-Ho Tsang, "Review: Faster R-CNN (Object Detection)," Towards Data Science, 2018. [Online]. Available: https://towardsdatascience.com/review-faster-r-cnn-object-detection-f5685cb30202. [Accessed: 10-April-2019].
21. G. Song, Y. Liu, M. Jiang, Y. Wang, J. Yan, and B. Leng, "Beyond Trade-Off: Accelerate FCN-Based Face Detector with Higher Accuracy," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7756–7764.
22. Venkata Krishna Jonnalagadda, "Object Detection YOLO v1 , v2, v3," 2019. [Online]. Available: https://medium.com/@venkatakrishna.jonnalagadda/object-detection-yolo-v1-v2-v3-c3d5eca2312a. [Accessed: 20-April-2019].
23. K. Yan, S. Huang, Y. Song, W. Liu, and N. Fan, "Face recognition based on convolution neural network," in 2017 36th Chinese Control Conference (CCC), 2017, pp. 4077–4081.
24. W. Yang and Z. Jiachun, "Real-time face detection based on YOLO," in 2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII), 2018, pp. 221–224.