Salient Object Detection using CNN

Nikunj Dhingra*,
Utkarsh Kumar**,
Abhishek Mittal***
Sarita Yadav****

Vivekananda Journal of Research
July - December 2020, Vol. 9, Issue 2, 177-190
ISSN 2319-8702(Print)
ISSN 2456-7574(Online)
Peer Reviewed Refereed Journal

http://www.vips.edu/vjr.php

Abstract

Advancement and enhancement in object detection was possible due to the growth and development of Convolutional Neural Networks(CNNs). When compared to classification problems, the object detection problems are more intensive and complex. Cross-platform object detection tasks are tedious and tortuous and so tackling problems involving salient object detection requires a novel and better approach. Here, various techniques in Object Detection using CNN are analysed which include R-CNN, Fast-CNN, YOLOv2 and MobileNetV2 SSD CNNs focusing primarily on the last two CNN architectures and compare their accuracies, speed, and various other factors.

Keywords: *CNN,R-CNN,YOLOv2,SSD,MAdd,mAP,VGG,ResNet,SVM,RoI,CHW,RELU,VOC, COCO-SSD*

Introduction

The ability of machine or computer to detect the objects in the physical world with image processing and classifying techniques or Object Detection, is done in order to gather and use the information in the virtual world for a wide variety of purposes ranging from alert and control systems to the development of autonomous robotic systems and futuristic machines.

* Department of Information Technology, Bharati Vidyapeeth's College of Engineering, New Delhi
Email: nikunjdhingra21@gmail.com

** Department of Information Technology, Bharati Vidyapeeth's College of Engineering, New Delhi
Email: utkarsh9799@gmail.com

*** Department of Information Technology, Bharati Vidyapeeth's College of Engineering, New Delhi
Email: officialabhishekmittal@gmail.com

**** Department of Information Technology, Bharati Vidyapeeth's College of Engineering, New Delhi
Email: sarita1320@yahoo.co.in

Needless to say, after 2012, the era of deep learning began and as discussed in [29], the advancements in deep neural networks and information processing helped CNNs to break all the benchmarks set by existing algorithms and became the gold standard for image processing, object localization and detection.

Though it is impractical to compare every object detection model, the goal of this paper is to compare the performance of R-CNN, Fast-CNN, YOLOv2 and MobileNetV2 SSD and how they stand when tested against some famous common data sets and analyze the tabulated results in the previous papers to compare their performance statistics and architectures.

Convolutional Neural Network

This field of study deals mostly with CNN based deep learning models which are pre-trained on an enormous ImageNet dataset which consists of 1.2 million images to train the model. ImageNet dataset generally acts as a standard model when it comes to classify images at large scale. To compare the existing models, the number of MAdd operations and mAP were computed.

Now, we discuss the models based on the differences in their architecture , parameters and number of operations involved within the different layers of the neural network.

A. **VGG:** This model was named so because it was developed by the Visual Graphics Group research team at oxford university. As proposed in [12], [15] the architecture consists of 3 X 3 convolutions which are stacked in sequence followed by max-pooling layers in between. To serve the purpose of feature extraction, these layers are followed by fully connected layers. The VGG models range from 13 layers to 19 layers. We have drawn the comparison between the models by using the VGG-19 consisting of approximately 140 million parameters and 19.5G MAdd operations .

B. **ResNet:** This architecture implements the idea of “Identity shortcut connection” which is also called Skip Connections. As mentioned in [12],[15], in this model, the input feature map is allowed to skip a few layers and the input feature map is passed as reference to the final layers. We have chosen ResNet-50 for the purpose of comparison. This model consists of 23.5M parameters with about 4G MAdd operations.

C. **R-CNN:** As discussed in [19], to bypass the problem of selecting a huge number of regions, Ross Girshick et al. proposed a method where we use selective search to extract just 2000 regions from the image and he called them region proposals. Therefore, now, instead of trying to classify a huge number of regions, you can just work with 2000 regions. These 2000 region proposals are generated using the selective search algorithm which is written below.

Selective Search:

1. Generate initial sub-segmentation, multiple candidate regions are generated.
2. Use greedy algorithm to recursively combine similar regions into larger ones
3. Use the generated regions to produce the final candidate region proposals.

These 2000 candidate region proposals are warped into a square and fed into a convolutional neural network that produces a 4096-dimensional feature vector as output. The CNN acts as a feature extractor and the output dense layer consists of the features extracted from the image and the extracted features are fed into an SVM to classify the presence of the object within that candidate region proposal. In addition to predicting the presence of an object within the region proposals, the algorithm also predicts four values which are offset values to increase the precision of the bounding box.

D. **Fast-CNN:** In [21], the author discussed the approach which is similar to the R-CNN algorithm. But, instead of feeding the region proposals to the CNN, the input image is fed into the CNN to generate a convolutional feature map. From the convolutional feature map, the regions of proposals are identified and warp them into squares and by using a RoI pooling layer, they are reshaped into a fixed size so that it can be fed into a fully connected layer. From the RoI feature vector, a softmax layer is used to predict the class of the proposed region and also the offset values for the bounding box.

The reason “Fast R-CNN” is faster than R-CNN is because you don’t have to feed 2000 region proposals to the convolutional neural network every time, as mentioned in [12],[21]. Instead, the convolution operation is done only once per image and a feature map is generated from it.

E. **MobileNetV2 SSD CNN:** As mentioned in [12],[15],[20],[22] , this architecture comprises depth-wise separable 3X3 convolutions in contrast with inverted ResNet architecture. One of the major differences between MobileNet and ResNet is that in ResNet the 3 X 3 convolutions are performed on reduced channels while in case of ResNet the 3 X 3 convolutions are completely replaced by depth-wise separable 3 X 3 convolutions along with an increase in the number of channels.

The ResNet architecture extracts features at 3 X 3 convolution on half of the input feature channel but in case of MobileNet, there is an increase in number of feature channels by an expansion factor 't', which can be found in [20],[22].

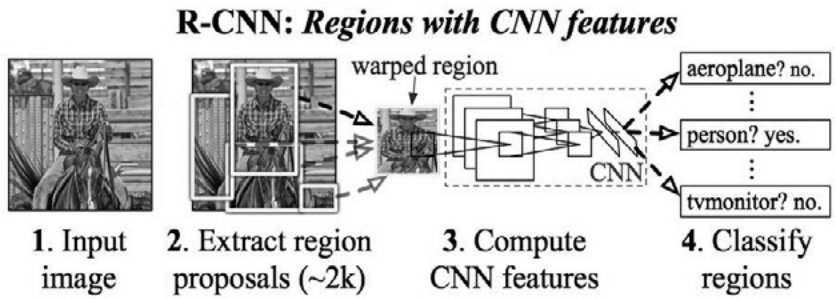

Figure 1. R-CNN

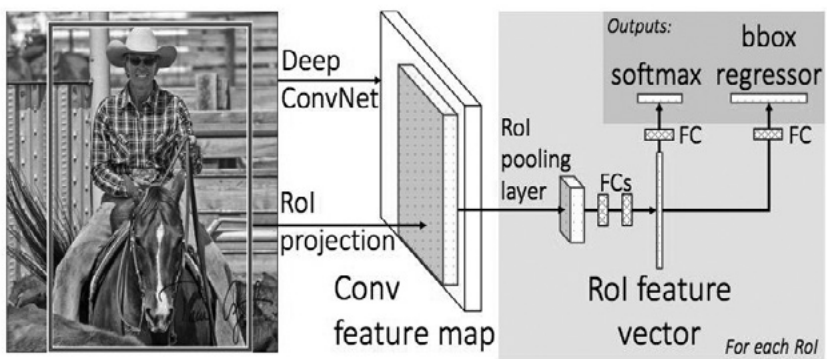

Figure 2. FAST CNN

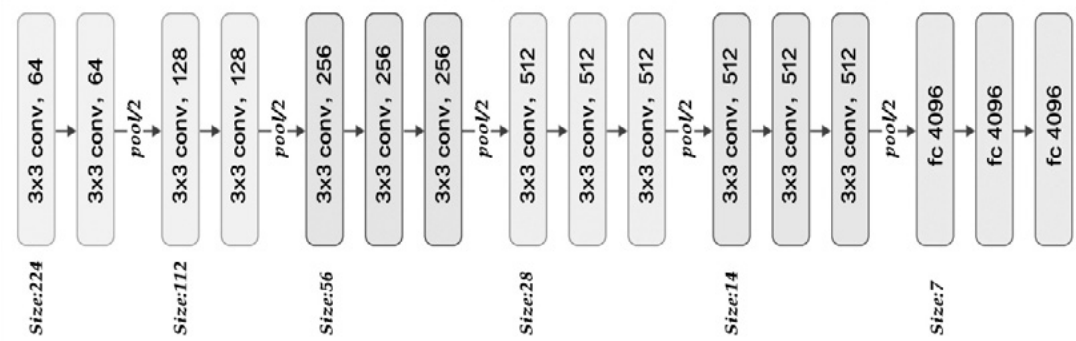

Figure 3. VGG-19

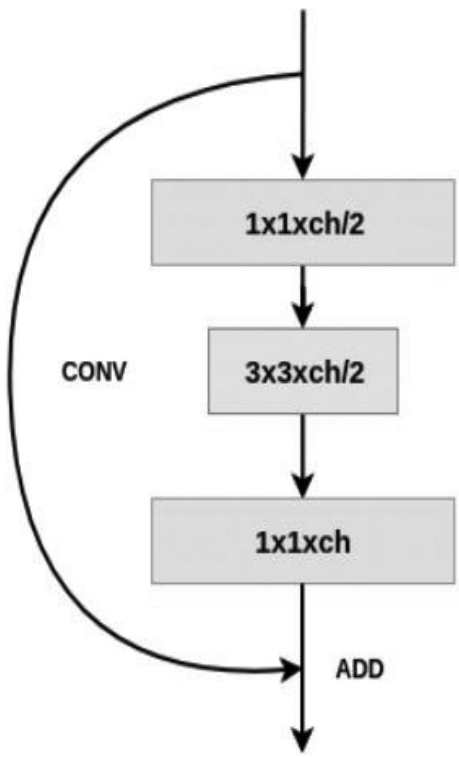

Figure 4. ResNet-50

Related Work

The existing research in the domain of Object Detection using CNN involves algorithms and techniques like SVMs, SSD, YOLO which can be found in [1], [4], [6], [8], [11] and [13]. In the SVM approach in [13], support vector machines were used as the algorithm to inspect the data for image classification and analysis. The SVM maps the data by analyzing and differentiating the features that can be compared, with the gap to be made as wide as possible. Then the classification can be either linear or nonlinear based on the kernel. The methods that are used in kernels can be used to analyze the patterns in images. Unsupervised models are needed for clustering the data into groups and when the data sets aren't labelled. These groups that are formed are the ones on which the data is mapped. The main drawback of this approach in [13], is the presence of residual points in the dataset while dividing similar points, and its computation speed as discussed in [13], [14].

The YOLO approach suggested in [1], [6] involves the application of a single neural network to the full image, which is then divided into regions and predicts bounding boxes and probabilities for each region with a predefined threshold to be considered as an object. As each grid cell can predict only a few bounding boxes and can only have one class, there exists strong spatial constraint on bounding boxes as discussed in [17] and [18]. So this may lead to incorrect localizations and therefore, YOLO being a fast approach, has its accuracy as a flaw which is quite notable for small objects.

SSD approach as proposed and discussed in [11] and [20], discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location [11].

The Single Shot Detector(SSD) overcomes the flaws of YOLO by doing the tasks of localization and classification in a single forward pass of the network. SSD being easy to train can incorporate tweaks like fixing priors and location loss improved the performance of SSD manifolds by making it highly accurate and precise.

Comparison

A. Design & Architecture

1. YOLOv2

Layer description	Layer output	
Network Input	416x416x3	
Convolutional, size=3, stride=1, pad=1, filters=16, bn=1	416x416x16	
Leaky ReLu	416x416x16	
Maxpool, size=2, stride=2	208x208x16	
Convolutional, size=3, stride=1, pad=1, filters=32, bn=1	208x208x32	
Leaky ReLu	208x208x32	
Maxpool, size=2, stride=2	104x104x32	
Convolutional, size=3, stride=1, pad=1, filters=64, bn=1	104x104x64	
Leaky ReLu	104x104x64	
Maxpool, size=2, stride=2	52x52x64	
Convolutional, size=3, stride=1, pad=1, filters=128, bn=1	52x52x128	
Leaky ReLu	52x52x128	
Maxpool, size=2, stride=2	26x26x128	
Convolutional, size=3, stride=1, pad=1, filters=256, bn=1	26x26x256	
Leaky ReLu	26x26x256	
Maxpool, size=2, stride=2	13x13x256	
Convolutional, size=3, stride=1, pad=1, filters=512, bn=1	13x13x512	
Leaky ReLu	13x13x512	
Maxpool, size=2, stride=1	13x13x512	
Convolutional, size=3, stride=1, pad=1, filters=1024, bn=1	13x13x1024	
Leaky ReLu	13x13x1024	
Convolutional, size=3, stride=1, pad=1, filters=B(5+C), bn=0	13x13xB(5+C)	← Network Output

Figure 5. YOLOv2

The architecture for the YOLOv2 can be visualized in the figure above. The details of each block in the visualization can be seen by hovering over the block. As mentioned in [24],[25], each Convolution block has the Batch Norm normalization and then Leaky Relu activation except for the last Convolution block. The Reorg layer after the Conv13_512 (refer above diagram) is a reorganization layer. If the input image has dimension 3x416x416 (CHW), then Conv13_512 would have an output size of 512x26x26 (CHW). The reorganization layer takes every alternate pixel and puts that into a different channel. Let us take an example of a single channel with 4x4 pixels as shown below. The reorganization layer reduces the size to half and creates 4 channels with adjacent pixels in different channels. Hence, the output of the Reorg layer from Conv13_512 will be 2048x13x13.Particularly in YOLOv2, the shape of output is 13x13xD, where D varies depending on the number of classes to be detected (D=5 for single class). The first 2-dimensional array (13x13) is called **grid cells**. So, there are 169 grid cells in total. One grid cell is 'responsible' for detecting 5 bounding boxes, that is up to 5 boxes can be detected on a grid cell. This means that the network can detect up to 169 x 5 = 845 boxes at once. This number of bounding boxes a grid cell can detect is actually the number of Anchor-Boxes that are prepared, and this number can be changed as per the requirements.

2. MobileNet V2 CNN

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Figure 6. MobileNet V2 CNN table

As mentioned in [15],[20],[22], the basic building block of the MobileNet V2 CNN is a bottleneck with depth-wise separable convolutions along with inverted residuals.

Each line describes a sequence of 1 or more identical (modulo stride) layers, repeated n times. All layers in the same sequence have the same number c of output channels. The first layer of each sequence has a stride s and all others use stride 1. All spatial convolutions use 3×3 kernels.

The expansion factor 't' is always applied to the input size as described in Table below:

Input	Operator	Output
$h \times w \times k$	1x1 conv2d , ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3x3 dwise s=s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

Figure 7. Application of expansion factor

(Bottleneck residual block transforming from k to k 0 channels, with stride s, and expansion factor t)

The architecture consists of initial fully convolutional layers with 32 filters followed by 19 residual bottleneck layers. RELU6 is used due to its robustness in case of low-

precision computation. 3 X 3 kernels is used as a standard size for the CNN. Apart from the first layer, the expansion factor is kept constant.

The primary network (width multiplier 1, 224 × 224) has a computational cost of 300 million MAdd operations with approximately 3.4 million parameters whereas in general, the number of parameters in the model can vary between 1.7M and 6.9M.

B. Performance

The traditional CNN approaches for object detection like Fast CNN, Masked CNN, R-CNN, all fell short in terms of both the speed and the accuracy against the more recent YOLOv2 and MobileNetV2 SSD CNNs as shown in [1], [11], [20] and [26] which are used for cross platform, real time and accurate object detection with less computational cost. So, our main focus is to compare the speed and accuracy of the YOLOv2 and MobileNetV2 SSD CNN.

It wouldn't be best to be comparing the results from different papers but is suitable for a bigger picture on where they stand. So, in [11] and [26] the models were compared by testing them on the Pascal VOC 2007,2012 and the MS COCO datasets and tabulated them against mAP or the measuring accuracy and FPS.

1. MobileNetV2 SSD

Method	VOC2007 test		VOC2012 test		COCO test-dev2015 trainval35k		
	07+12	07+12+COCO	07++12	07++12+COCO			
	0.5	0.5	0.5	0.5	0.5:0.95	0.5	0.75
SSD300	74.3	79.6	72.4	77.5	23.2	41.2	23.4
SSD512	76.8	81.6	74.9	80.0	26.8	46.5	27.8
SSD300*	77.2	81.2	75.8	79.3	25.1	43.1	25.8
SSD512*	**79.8**	**83.2**	**78.5**	**82.2**	**28.8**	**48.5**	**30.3**

Figure 8.1 Results of Pascal VOC 2007, 2012 and MS COCO

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

Figure 8.2 Performance comparison

2. YOLOv2 on Pascal VOC 2007 dataset

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

Figure 9. YOLOv2 on Pascal VOC 2007

3. YOLOv2 on Pascal VOC 2012 dataset

Method	data	mAP
Fast R-CNN [5]	07++12	68.4
Faster R-CNN [15]	07++12	70.4
YOLO [14]	07++12	57.9
SSD300 [11]	07++12	72.4
SSD512 [11]	07++12	74.9
ResNet [6]	07++12	73.8
YOLOv2 544	07++12	73.4

Figure 10. YOLOv2 on Pascal VOC 2012

4. YOLOv2 on MS COCO dataset

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [16]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [20]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [34]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [32]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [27]	DarkNet-19 [27]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [22, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet (ours)	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2

Figure 11. YOLOv2 on MS COCO

So, from the above results in [11] and [26], it can be concluded that MobileNetV2 SSD has a very high FPS and accuracy, but its performance dips sharply for smaller objects while the YOLOv2 does take less time but its accuracy and precision is slightly lesser than that of MobileNetV2 SSD.

C. Feature Extractor

The higher performance on COCO can be correlated with the higher performance on classification. This is verified by the author in [12] by investigating the relationship between overall mAP of different models and Top-1 Imagenet classification accuracy attained by pre-trained feature extractor used to initialize each model. This correlation was only significant for YOLOv2 while the performance of SSD appears to be less reliant on its feature extractor's classification accuracy.

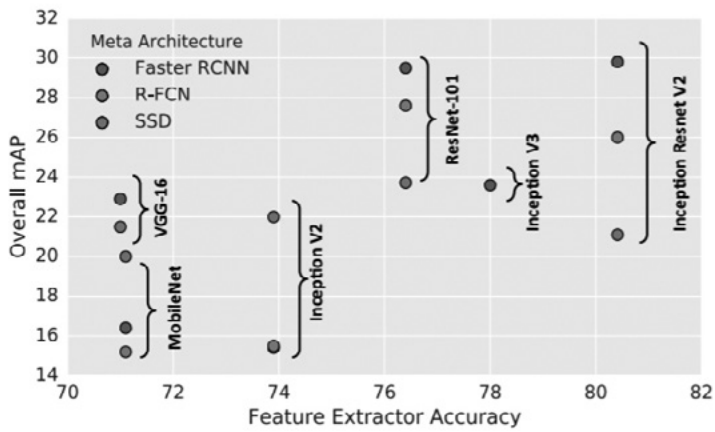

Figure 12. Feature extractor accuracy

D. Object Size and Image Resolution

For large objects, MobileNetV2 SSD can perform pretty well even with a simple extractor as discussed in [20] and [12]. SSD can even match other detectors' accuracies using better extractor. But SSD performs much worse on small objects compared to other methods.

Through the results in [12] we can easily conclude that the higher the resolution of the image, the more the accuracy of detecting the objects, and a higher resolution increases the accuracy of detecting smaller objects significantly and accuracy of large objects slightly.

When decreasing resolution by a factor of two in both dimensions, accuracy is lowered by 15.88% on average but the inference time is also reduced by a factor of 27.4% on average.

Conclusion

On examining, assessing and through evaluation of the object detection with CNN techniques, and more specifically, the YOLOv2 and MobileNetV2 SSD CNN architectures, few comparisons were drawn on the basis of their architectures and design, their number of parameters, their ability to incorporate any tweaks and changes, their parameter count, and most importantly, its effect on the computational cost of the object detection model, i.e., the speed and the precision or accuracy of the model. Based on the comparisons that were drawn, MobileNetV2 CNN was found to be a better CNN based on speed and performance. The average inference time for MobileNetV2 CNN is less than that for YOLOv2 and since the resolution of the input image is a key factor for object detection, MobileNetV2 CNN has better performance and accuracy with high resolution images when compared to YOLOv2.

References

Joseph Redmon, Santosh Divvala, Ross Girshick, "You Only Look Once: Unified, Real-Time Object Detection", The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788.

Girshick, R.: Fast R-CNN. In: ICCV. (2015).

Hoiem, D., Chod Pathumwan, Y., Dai, Q.: Diagnosing error in object detectors. In: ECCV 2012. (2012).

Akash Tripathi, T.V. Ajay Kumar, Tarun Kanth Dhansetty, J. Selva Kumar."Real Time Object Detection using CNN", International Journal of Engineering & Technology, 7 (2.24) (2018) 33-36.

Lu Zhang, Ju Dai, Huchuan Lu, You He, Gang Wang,"A Bi-directional Message Passing Model for Salient Object Detection", The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 1741-1750.

F. Particke1,*, R. Kolbenschlag, M. Hiller, L. Patiño-Studencki1 and J. Thielecke,"Deep Learning for Real-Time Capable Object Detection and Localization on Mobile Platforms",IOP Conf. Series: Materials Science and Engineering 261 (2017) 012005, AIAAT 2017.

Geethapriya. S, N. Duraimurugan, S.P. Chokkalingam,"Real-Time Object Detection with Yolo",International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-8, Issue-3S, February 2019.

Jiang-Jiang Liu, Qibin Hou, Ming-Ming Cheng, Jiashi Feng, Jianmin Jiang,"A Simple Pooling-Based Design for Real-Time Salient Object Detection",cs.CV(2019).

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

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich,"Going Deeper with Convolutions",The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1-9.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg,"SSD: Single Shot MultiBox Detector," cs.CV(2016).

Jonathan Huang Vivek Rathod Chen Sun Menglong Zhu Anoop Korattikara Alireza Fathi Ian Fischer Zbigniew Wojna Yang Song Sergio Guadarrama and Kevin Murphy, "Speed/ accuracy trade-offs for modern convolutional object detectors", cs.CV(2017).

G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Fuzzy SVM for 3D facial expression classification using sequential forward feature selection" Computational Intelligence and Communication Networks (CICN), 2017 9th International Conference. 16-17 Sept. 2017.

T. Ahonen, A. Hadid and M. Pietikainen, "Face Description with Local Binary Patterns: Application to Face Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037-2041, Dec. 2006.

Narsi Reddy, Ajita Rattani, Reza Derakhshani,"Comparison of Deep Learning Models for Biometric-based Mobile User Authentication."

Redmon J and Farhadi A. Yolo9000: Better, faster, stronger. arXiv preprint arXiv:1612.08242, 2017.

Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C, and Berg A. Ssd: Single shot multibox detector. In European Conference on Computer Vision, pages 21–37. Springer, 2016.

Redmon J, Divvala S, Girshick R, and Farhadi A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 779–788, 2016.

Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik," Region-based Convolutional Networks for Accurate Object Detection and Segmentation."

Mark Sandler, Andrew Howard, Mangling Zhu, Andrey Zhmoginov, Liang-Chieh Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks."

Ross Girshick, "Fast R-CNN ", Microsoft Research, computer vision foundation.

Andrew G. Howard,Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam,"MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications",[csCV] April, 2017.

Robert J. Wang, Xiang Li, Charles X. Ling,"Pelee: A Real-Time Object Detection System on Mobile Devices."

Jun Sang, Zhongyuan Wu, Pei Guo, Haibo Hu, Hong Xiang, Qian Zhang, Bin Cai,"An Improved YOLOv2 for Vehicle Detection",MDPI, December 2018.

Jianming Zhang, Manting Huang, Xiaokang Jin,Xudong Li,"A Real-Time Chinese Traffic Sign Detection Algorithm Based on Modified YOLOv2",MDPI November, 2017.

Joseph Redmon, Ali Farhadi, University of Washington, Allen Institute for AI.

S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside Outside net: Detecting objects in context with skip pooling and recurrent neural networks. arXiv preprint arXiv:1512.04143, 2015

J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. arXiv preprint arXiv:1605.06409, 2016.

J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le, et al. Large scale distributed deep networks. In Advances in neural information processing systems.