Understanding of Object Detection Based on CNN Family and YOLO

To cite this article: Juan Du 2018 *J. Phys.: Conf. Ser.* **1004** 012029

IOP Conf. Series: Journal of Physics: Conf. Series **1004** (2018) 012029 doi:10.1088/1742-6596/1004/1/012029

Understanding of Object Detection Based on CNN Family and YOLO

Juan Du[1,*]

[1]New Research and Development Center of Hisense, Qingdao 266071, China

*dqxwpl@sina.com

Abstract. As a key use of image processing, object detection has boomed along with the unprecedented advancement of Convolutional Neural Network (CNN) and its variants since 2012. When CNN series develops to Faster Region with CNN (R-CNN), the Mean Average Precision (mAP) has reached 76.4, whereas, the Frame Per Second (FPS) of Faster R-CNN remains 5 to 18 which is far slower than the real-time effect. Thus, the most urgent requirement of object detection improvement is to accelerate the speed. Based on the general introduction to the background and the core solution CNN, this paper exhibits one of the best CNN representatives You Only Look Once (YOLO), which breaks through the CNN family's tradition and innovates a complete new way of solving the object detection with most simple and high efficient way. Its fastest speed has achieved the exciting unparalleled result with FPS 155, and its mAP can also reach up to 78.6, both of which have surpassed the performance of Faster R-CNN greatly. Additionally, compared with the latest most advanced solution, YOLOv2 achieves an excellent tradeoff between speed and accuracy as well as an object detector with strong generalization ability to represent the whole image.

1. Introduction

Humans glance at an image and instantly know what objects are in the image, where they are, and how they interact. If algorithms for image processing could be accurate and fast enough, the computers would be able to drive cars without specialized sensors, and assistive devices would be able to convey real-time scene information to users. Likewise, if these algorithms could complete Deep Learning (DL) tasks with high efficient and excellent performance like human beings do, it would be real Artificial Intelligent (AI). Thus, the core tasks of image processing are a serial of recognition: classification, localization and object detection; and the key challenges are: accuracy, speed, cost and complexity.

To achieve an ideal effect, algorithms of recognition have gone a long way before the significant boom of Convolutional Neural Network (CNN) brought by AlexNet [1] in 2012. From the early concept of the receptive fields [2] and the improved Neural Network (NN) models (Rosenblatt [3] in 1962; Kabrisky [4] in 1966; Giebel [5] in 1971; Fukushima [6] in 1975) to Back Propagation Neural Netowrk (BP_NN) [7, 8] in 1986. The convolutional framework came out in Fukushima's Neocognitron [9] in 1980s, a biologically inspired hierarchical and shift-invariant pattern recognition model with shortage of lacking supervised training algorithm. It first grew into classical model LeNet [10] in 1998. CNNs saw heavy use in 1990s [11] but had been overshadowed by shallow machine learning models (eg. Support Vector Machines (SVM), Boosting, Maximum Entropy Methods). Until 2012, AlexNet via CNN reduced the error rate from 26% to 15.3%, which dramatically fueled the development of DL, rekindled CNNs and GPU.

To approach higher accuracy and speed, CNNs have been developed in two directions after being popular in 2012. One direction is optimization, such as batch normalization, VGGNet [12] (22 layers; Error rate is 7.3%) of Visual Geometry Group (VGG), GoogLeNet [13] (22 layers, error rate is 6.67%) which gained the champion of ImageNet Large Scale Visual Recognition Challenge 2014 (2014 SVRC), and Deep Residual Learning [14] (ResNet, 152 layers, error rate is 3.57%). The other direction is to improve the framework of image processing. CNN's strong ability of feature extraction and classification has good effect on dealing with image with single object, while the problem of window selection was highlighted when the image contains multi-objects. To solve the problem, Regions with CNN (R-CNN) introduced new techniques, such as region proposals, crop/warp, SVM classification and bounding-box regression. R-CNN achieved excellent object detection accuracy with the Mean Average Precision (mAP) of 54% on Visual Object Classes (VOC) 2010 compared to 33% for the Deformable Part Model (DPM) [15, 16] which is based on Histogram of Oriented Gradients (HOG). However, R-CNN's pre-extraction and multi-stage pipeline using cross-entropy loss for each object proposal consume expensive space and time. Then, with the help of warping proposal behind the R-CNN's last convolutional layer, Spatial Pyramid Pooling Network (SPPNet) [17] accelerated its test speed by 10 times and training speed by 3 times than those of R-CNN. And SPPNet shared similar drawbacks with R-CNN as its multi-stage pipelines. Fast R-CNN solved the problems with a single-stage and multi-task loss; added Region of Interest (ROI) to enhance the whole training; and used Singular Value Decomposition (SVD) algorithm to speed up. Fast R-CNN advanced with 10 times' test speed and 3 times' training speed than SPPNet, gained mAP value of 70. Even though Fast R-CNN seemed to have achieved nearly real-time rates using very deep networks [12]. Its proposals are the computational bottleneck in detection systems as well. Therefore, Faster R-CNN effectively introduced Region Proposal Networks (RPN) to share convolutional layers with object detection networks and computed proposals with deep net. It gained a nearly cost-free computation of the detection network. Faster R-CNN with ResNet [18] gained mAP value of 76.4 and Frame Per Second (FPS) of 5. On the accuracy of image recognition, Faster R-CNN has indeed achieved fairly good effect, while dealing speed of Faster R-CNN still has space to quicken in the state-of-the-art detection systems.

In this paper, a new direction of solving the bottleneck of proposals You Only Look Once (YOLO) will be discussed after the background descriptions of CNNs (especially the key incarnation Faster R-CNN). Two versions of YOLO will be introduced: YOLO V1 and V2; YOLO's layers, algorithms and characteristics will all be listed. By comparing YOLO with Faster R-CNN on accuracy, speed, cost and complexity, their advantages and shortages would be exposed. At last, the conclusion of YOLO and Faster R-CNN will be summarized with the outlook to the CNN future.

2. The Introduction of CNN

CNN [19] started with LeNet in 1998 and prospered in 2012 because of AlexNet, with the error rate 15.3%. Then, ZF-net [20], GoogLeNet and VGGNet brought lower and lower error rate. By the end of 2015, ResNet has gained an error rate 3.6% which was even less than that of the human eyes (5.1%). And this is the first time that the recognition capability of deep learning models has surpassed that of the human beings.

2.1. Layers of CNN

The framework of the network layers was kept from NN to CNN, while the function and formation of the network have been changed. The layers of CNN are shown as following.

2.1.1. Input Layer The main task of input layer is to initialize the input image data to make all the dimensions of the input data zero-centered. Then, it normalizes the scale of all the input data within [0,1] to accelerate the converging speed, and whitens the data to decrease the redundancy. It utilizes Principal Components Analysis (PCA) to degrade and decorrelate the dimensions of the data to make them focused on just several key factors [1].

2.1.2. Convolutional (CONV). Layer Convolutional layer is the core of CNNs. It uses CONV kernel as a filter to slide on the original image. The values of each pixel of the local correlated data within the filter will be multiplied and added as the convolutional results. This type of rule is called convolution which enables the features of the image extracted with designed CONV kernel. Besides, the method of filtering different parts of an image with same CONV kernel is called shared weight, which enables that the neutral cells with same feature can be recognized and classified into same object type. The parameters include: kernel size, depth, stride, zero-padding, and filter quantity. The calculation algorithm of the output size is:

$$\mathbf{H_{out}} = \mathbf{1} + \frac{\mathbf{H_{in}+(2*pad)-K_{height}}}{\mathbf{S}} \;;\quad \mathbf{W_{out}} = \mathbf{1} + \frac{\mathbf{W_{in}+(2*pad)-K_{width}}}{\mathbf{S}} \tag{1}$$

2.1.3. Active Layer. To make the result of the CONV layer nonlinear, active Layer is set to solve the vanishing gradient problem caused by the underfitting. There are many functions for it: Sigmoid, Tanh, the rectified Linear Unit (ReLU), Leaky ReLU, the Exponential Linear Unit (ELU), and Maxout. Recently, Leaky ReLU is the most popular as it converges faster than Sigmoid and Tanh, calculates in a simpler and more efficient way, and contains no dead area.

2.1.4. Pooling Layer. Pooling layer is used to decrease the dimension of the results from CONV layer, located between two CONV layers. There are three types of pooling: general pooling, overlapping pooling and Spatial Pyramid Pooling (SPP). The general pooling's width is often as the same as stride. It's typical ways are max pooling and average pooling. The overlapping pooling normally has longer width than the stride. SPP can transform the convolutional features of images with any size, into the same number of dimensions. This advantage of SPP enables that CNN could not only deal with multi types of images but also avoid the information loss caused by cropping and warping, which makes it the core of SPPNet.

2.1.5. Fully Connected Layer. Fully connected layers are often the last layer at the end of CNNs. They transmit the data to the output, as well as simplify and speed up the data calculation.

2.1.6. Other Layers. Some of CNNs models still have dropout layers and regression layers. The former layer is to solve the overfitting. To avoid the weight is too subjective, it typically updates the weight of the neural cell knot with a certain probability decided by the stochastic policy. Regression layer is to classify the features. There are many methods: Logistic Regression (LR), Gaussian Processes for Regression (GPR), Bayesian Linear Regression (BLR). The regression gives the probabilities of all the possible object types, which is the base of final judgement.

2.2. Advantages and Shortages of CNN

CNN offers an effective mathematical way to express and extract the features of the input data. The weight sharing mechanism can recognize the data with same feature, which enables CNNs dealing with high dimensions' data and laying foundation for the final excellent classification. Its pooling layer reduces the dimensions easily. These all make CNNs very competitive and flourishing on intelligent recognition in many areas of data processing. However, to get better results, the main barrier is the scope of the parameters and datasets, as well as the computing capability of the hardware. These high physical requirements and CNN's obscure physical meaning are all current challenges.

3. CNN Family

The development process of CNN is to build more stable and more efficient system for object detection. As along with the definition of proposal, Region with CNN, Fast Region with CNN (Fast R-CNN), and Faster Region with CNN (Faster R-CNN) have been proposed. This section is going to introduce these algorithms in detail.

3.1. R-CNN

R-CNN [21] is an important milestone of CNN for object detection. It is the first neural network propounding the region proposal to realize the object detection based on the excellent ability of CNN's feature extraction and classification. The region proposal gets the potential objects by sliding the proposals with different width and height, such as selective search. Before CNN, R-CNN proposes crop/warp to normalize the candidate images with fixed size to standard. After CNN, it adds SVM classification and bounding-box regression to get the exact object detection result. But region proposal makes R-CNN consume massive data, time, computation and energy; crop/warp brings information loss. Both of them leave space for further improvement to the R-CNN.

3.2. Fast R-CNN
Fast R-CNN [22] is proposed based on SPPNet. SPPNet deletes the crop/warp of R-CNN, replaces the last pooling layer before FC layer with SPP, and keeps the output image m*n parts no matter what resolution of the input image is. These features accelerate the test speed by 24 to 102 times. Fast R-CNN profferes Region of Interest (ROI) pooling and proposal reflection, which solve the whole training problem of SPP. Besides, it uses multi-task loss layer: SoftmaxLoss replaces SVM, and SmoothL1Loss replaces Bouding-box. These new methods raise the precision of the algorithm by combining the classification and regression. Additionally, it speeds up the fully connected layers with SVD. Therefore, the training and test speed of Fast R-CNN are respectively 3 times and 10 times faster than that of SPP. VOC07 dataset shows the mAP of Fast R-CNN is 70.

3.3. Faster R-CNN
Based on Fast R-CNN, Faster R-CNN [23] solves the regional proposal problem by adding RPN, which is the Key contribution of Faster R-CNN. It gets the regional proposal not on the original image but the final feature image which will be input into the ROI pooling. As the resolution of the feature image is lower than that of the original image, the computation of Faster R-CNN is sure much less than that of all the former models of CNN.

The core feature of Region Proposal Network (RPN) is to get the proposal by sliding it. Each sliding proposal will generate 9 candidate anchors with different scales, widths and heights. Faster R-CNN's course of extracting the anchors' features is similar with that of the Fast R-CNN, while its object classification is only to recognize that the features are foreground or background. And its proposal regression is only to find out a more precise location of the target object. For each location of the proposal, RPN uses two full-connected layers (object classification and proposal regression) to judge and abandon the anchors. It never does explicit regional proposal. The main rules of selecting the anchors are: (1) dismissing the anchors on the boundary; (2) the anchors whose overlapping area with the sample are greater than 0.7 would be classified as foreground, and those whose overlapping areas are smaller than 0.3 would be classified to be background. In this way, RPN chooses about 300 anchors for each sliding proposal.

Faster R-CNN uses the alternating training mode to train the shared features. It uses the recent network to initiate the weight of the RPN, extracts the right proposals from the training dataset and trains the Faster R-CNN model with the proposals repeatedly until the result converges well.

Faster R-CNN has already offered a result with perfect precision of recognition. Its only left space for further improvement is speed, which is the main reason for the later algorithms doing.

4. YOLO
A new approach to object detection is called You Only Look Once (YOLO) which means that an image can be predicted what the objects are and where they are at one glance. As the first method entirely throwing out pipeline, it frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities, which are predicted with a single neural network from full images in one evaluation. Besides, YOLO selects GoogLeNet but not VGG-16 as base network, as their rates of the forwarding transport computation versus precision are$\frac{852}{88}$ VS $\frac{3069}{90} \approx \frac{1}{3.52}$, the former is far faster than VGG. So, YOLO is fast by design and indeed real-time while keeping

good accuracy.

4.1. YOLOv1

Base YOLO [24] is also called YOLO Version 1 (YOLOv1). It models detection as a regression problem. A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes.

YOLOv1 divides the input image into an $S \times S$ grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes, confidence scores for those boxes and C class probabilities of the grid. These predictions are encoded as an $S \times S \times (B \times 5 + C)$ tensor. In the test process, YOLOv1 multiplies the conditional class probabilities and the individual box confidence predictions which give us class-specific confidence scores for each box:

$$\mathbf{Pr}(\mathbf{Class_i}|\mathbf{Object}) \times \mathbf{Pr}(\mathbf{Object}) \times \mathbf{IOU}_{\mathbf{pred}}^{\mathbf{truth}} = \mathbf{Pr}\ (\mathbf{Class_i}) \times \mathbf{IOU}_{\mathbf{pred}}^{\mathbf{truth}} \quad \mathbf{(2)}$$

(IOU stands for Intersection Over Union). The scores encode both the probability of that class appearing in the box and how well the predicted box fits the object. Each bounding box consists of 5 predictions: x, y, w, h and confidence. The(x, y) coordinates represent the center of the box relative to the bounds of the grid cell. The width w and the height h are predicted relative to the whole image. And that is why YOLOv1 uses $B \times 5$ for calculating tensor. For evaluating YOLO on Pascal VOC, $S = 7, B = 2$ are normally used. Pascal VOC has 20 labelled classes, so $C = 20$. YOLOv1's final prediction is a $7 \times 7 \times (5 \times 2 + 20) = 7 \times 7 \times 30$ tensor. It uses only 98 bounding boxes per image versus 2000 from Selective Search.

YOLOv1's network has 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules used by GoogLeNet, YOLOv1 simply uses 1×1 reduction layer followed by 3×3 convolutional layers, similar to Lin et al [25]. On Pascal VOC2007, YOLOv1 processes images at 45 frames per second (FPS) which is twice to nine times faster than that of Faster R-CNN. Especially, Fast YOLO, a fast version of YOLO designed to push the boundaries of fast object detection, has reaches an astounding 155 FPS.

4.2. YOLOv2

YOLO Version 2 (YOLOv2) [18] is a variously improved model to YOLO which keeps the advantage on speed and try to rise the mAP value from YOLOv1's 63.4. Using a novel, multi-scale training method the same YOLOv2 model can run at varying sizes, offering an easy tradeoff between speed and accuracy.

YOLOv2 adds a list of significant solutions to increase mAP. Batch Normalization preprocesses the input data. High Resolution Classifier from YOLOv1's 224×224 to 448×448 raises the mAP by 4%. YOLOv2 also uses a new network with the thought "network in network", predicts with global average pooling, and compresses the features by putting 1×1 convolutional core among between the 3×3 convolutional cores. Its neural network has fewer convolutional layers (19 instead of YOLOv1's 24) and fewer filters plus 5 max pooling layers. Besides, YOLOv2 adopts convolutional with anchor boxes and increases each grid's resolution of from YOLOv1's 7×7 to 13×13. It has only 1 bounding box for each grid. Additionally, after researching the IOU function curve by counting the Ground Truth statistics on VOC and COCO with the K-means algorithm, YOLOv2 finds a solution to the best quantity of anchor boxes "Dimension Clusters" and final decides the number of anchor boxes to be 5, whose Average IOU (61.0) is tantamount with Faster R-CNN's 60.9 with 9 anchor boxes. Meanwhile, direct location prediction is also used to mitigate the instability brought by the anchor boxes, and it finally raises the mAP by 5% together with the good effect of dimension clusters. Finally, YOLOv2 adds a pass-through layer to get the extracted features from the former 26×26 layer and combine them with the original final output features, so that the ability of detecting the small object would be enhanced. In this mean, YOLOv2 raises the mAP by 1%.

4.3. Advantages and Shortages of YOLO
YOLO breaks through the max-speed limit of CNN and realizes excellent balance of speed and accuracy. It is extremely fast and real time. YOLOv2 has achieved 76.8 mAP at 67 FPS and 78.6 mAP at 40 FPS, outperforming state-of-the-art methods like Faster R-CNN with ResNet while still running precisely. Secondly, YOLO reasons globally and encodes contextual information about the image. Thus, it's less likely to predict false positives on background. Finally, YOLO learns very general representations of objects, outstripping other detection methods including DPM and R-CNN by a wide margin, when generalizing from natural images to other domains like artwork.

However, YOLO has limitations. It imposes strong spatial constraints on bounding box predictions such as recognizing small objects in group. It still struggles to generalize to objects in new or unusual aspect ratios or configurations. It's also not perfect that YOLO's loss function treats errors the same in small bounding boxes versus large boxes.

5. The Compare of Faster R-CNN and YOLOv2
YOLO suggests a completely new way to process image, varying too much from not only Faster R-CNN but also R-CNN and all its variants. Here only the key differences between YOLO and Faster R-CNN will be discussed as following.

5.1. The Framework
Though both Faster R-CNN and YOLO use CNN as core and their key purposes are all to find a better way of dividing proposals based on CNN, their frameworks differ from each other greatly. Faster R-CNN keeps the traditional general frame of R-CNN: using CNN dealing with the whole input image at first and dividing proposals later, maintaining regional proposal and ROL pooling. But its key contribution is adopting Regional Proposal Network to specially quicken the computation of processing proposals. YOLO divides the whole image at the very beginning and later uses CNN to process. Besides, YOLO gives up sliding windows and regional proposal completely and divides the input image into $S \times S$ grids. Meanwhile, it innovates confidence IOU mechanism to each grid and class probabilities for making decision. Though YOLOv2 reuses anchor boxes, YOLOv1 did give up it at the inception of YOLO; and YOLOv2 uses 5 anchor boxes while Faster R-CNN has 9 anchor boxes. In addition, YOLOv2 creates many new techniques to improve precision, such as dimension clusters, these are all what Faster R-CNN doesn't support.

5.2. Outside Performance
Faster R-CNN focuses on speeding up the R-CNN framework by sharing computation and using neural networks to propose regions instead of Selective Search. While they offer speed and accuracy improvements over R-CNN, both still fall short of real-time performance. Instead of trying to optimize individual components of a large detection pipeline, YOLO throws out the pipeline entirely and is fast by design. So YOLOv2 can achieve high precision and keep real-time for pictures with high resolution. While for pictures with low resolution, it shows outstanding high speed with excellent mAP value. Additionally, none of the detection systems has surpassed Fast YOLO's unparalleled performance with 155 FPS and mAP value 52.7 until now.

Table 1. The Performance Compare of the Detection Systems.

		Detection Frameworks	**Train**	**mAP ↑**	**FPS**	**PS (mAP×FPS)**	**PS Order**
Less Than Real-Time Detectors	1	Fastest DPM [26]	2007	30.4	15	456	11
	2	R-CNN Minus R [27]	2007	53.5	6	321	13
	3	Faster R-CNN ZF[20]	2007+2012	**62.1**	**18**	1118	9
	4	YOLO VGG-16[24]	2007+2012	**66.4**	**21**	1394	8
	5	Fast R-CNN[22]	2007+2012	70.0	0.5	35	14

	6	Faster R-CNN VGG-16[20]	2007+2012	73.2	7	512	10
	7	Faster R-CNN ResNet[20]	2007+2012	76.4	5	382	12
Real-Time Detectors	1	Fast YOLO [24]	2007+2012	52.7	155	8169	1
	2	YOLO(YOLOv1)[24]	2007+2012	63.4	45	2853	7
	3	YOLOv2 288×288[24]	2007+2012	69.0	91	6279	2
	4	YOLOv2 352×352[24]	2007+2012	73.7	81	5970	3
	5	YOLOv2 416×416[24]	2007+2012	**76.8**	**67**	5146	4
	6	YOLOv2 480×480[24]	2007+2012	**77.8**	**59**	4590	5
	7	YOLOv2 544×544[24]	2007+2012	**78.6**	**40**	3144	6

Table 1 shows theadvancement of YOLO. The index PS is actually created to show the general valueof each detection system based on regarding the precision and speed have same weight.PS is only for reference; its definition can be changed with different weight scale formAP and FPS. The weight scale shows the relative importance of mAP and FPS to target.

6. Conclusion

This paper briefly discusses about the current algorithms in object detection, including the CNN family and YOLO. Compared with CNNs, YOLO has more advanced application in practice. YOLO is a unified object detection model. It's simple to construct and can be trained directly on full images. Unlike classifier-based approaches, YOLO is trained on a loss function that directly corresponds to detection performance and the entire model is trained jointly. Fast YOLO is the fastest general-purpose object detector. And YOLOv2 provides state-of-the-art the best tradeoff between real-time speed and excellent accuracy for object detection than other detection systems across a variety of detection datasets. Furthermore, YOLO's better generalizing representation of object than other models making it ideal for applications that rely on fast, robust object detection. These preeminent and precious advantages make it worthy of being strongly recommended and popularized. Except the structure of each algorithm, the most urgent up-coming challenge for machine learning is the scope of the dataset. The availability of suitable training data could be the vital part in the learning process, to achieve an idea results.

Reference

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).
[2] Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of physiology, 160(1), 106-154.
[3] Rosenblatt, F. (1962). Principles of neurodynamics.
[4] Kabrisky, M. (1964). a Proposed Model for Visual Information Processing in the Human Brain.
[5] Giebel, H. (1971). Feature extraction and recognition of handwritten characters by homogeneous layers. In Zeichenerkennung durch biologische und technische Systeme/Pattern Recognition in Biological and Technical Systems(pp. 162-169). Springer, Berlin, Heidelberg.
[6] Fukushima, K. (1975). Cognitron: A self-organizing multilayered neural network. Biological cybernetics, 20(3-4), 121-136.
[7] Rumelhart, D. E.,Hinton,G. E.,&Williams,R.J.(1985). Learning internal representations by error propagation (No.ICS-8506). California Univ San Diego La Jolla Inst for Cognitive Science.
[8] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4), 541-551.

[9] Fukushima, K., & Miyake, S. (1982). Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In Competition and cooperation in neural nets(pp. 267-285). Springer, Berlin, Heidelberg.
[10] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4), 541-551.
[11] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.
[12] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
[13] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).
[14] He,K.,Zhang,X.,Ren,S.,&Sun,J.(2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision & pattern recognition (pp. 770-778).
[15] Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence, 32(9), 1627-1645.
[16] Girshick, R. B., Felzenszwalb, P. F., & McAllester, D. (2012). Discriminatively trained deformable part models, release 5.
[17] Hosang, J., Benenson, R., Dollár, P., & Schiele, B. (2016). What makes for effective detection proposals?. IEEE transactions on pattern analysis and machine intelligence, 38(4), 814-830.
[18] Redmon, J., & Farhadi, A. (2016). YOLO9000: better, faster, stronger. arXiv preprint arXiv:1612.08242.
[19] Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). CNN features off-the-shelf: an astounding baseline for recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 806-813).
[20] Ren, S., He, K., Girshick, R., & 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).
[21] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).
[22] Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).
[23] Ren, S., He, K., Girshick, R., & 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).
[24] Redmon, J., Divvala, S., Girshick, R., & 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).
[25] Lin, M., Chen, Q., & Yan, S. (2013). Network in network. arXiv preprint arXiv:1312.4400.
[26] Yan, J., Lei, Z., Wen, L., & Li, S. Z. (2014). The fastest deformable part model for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2497-2504).
[27] Lenc, K., & Vedaldi, A. (2015). R-cnn minus r. arXiv preprint arXiv:1506.06981.