International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 04 Issue: 07 | July -2020 ISSN: 2582-3930

Object Detection with Deep Learning

Utkarsh Mishra, Galgotias University final semester (B.tech c.s.e)

understanding, we should not only concentrate on classifying different images, but also try to precisely estimate the concepts and locations of objects contained in each image. This task is referred as object detection [1][S1], which usually consists of different subtasks such as face detection [2][S2], pedestrian detection [3][S2] and skeleton detection [4][S3]. As one of the fundamental computer vision problems, object detection is able to provide valuable information for semantic understanding of images and videos, and is related to many applications, including image classification [5], [6], human behavior analysis [7][S4], face recognition [8][S5] and autonomous driving [9], [10]. Meanwhile, Inheriting from neural networks and related learning systems, the progress in these fields will develop neural network algorithms, and will also have great impacts on object detection techniques which can be considered as learning systems. [11]–[14][S6]. However, due to large variations in viewpoints, poses, occlusions and lighting conditions, it's difficult to perfectly accomplish object detection with an additional object localization task. So much attention has been attracted to this field in recent years [15]–[18].

The problem definition of object detection is to determine where objects are located in a given image (object localization) and which category each object belongs to (object classification). So the pipeline of traditional object detection models can be mainly divided into three stages: informative region selection, feature extraction and classification.

Informative region selection. As different objects may appear in any positions of the image and have different aspect ratios or sizes, it is a natural choice to scan the whole image with a multi-scale sliding window. Although this exhaustive strategy can find out all possible positions of the objects, its shortcomings are also obvious. Due to a large number of candidate windows, it is computationally expensive and produces too many redundant windows. However, if only a fixed number of sliding window templates are applied, unsatisfactory regions may be produced.

Feature extraction. To recognize different objects, we need to extract visual features which can provide a semantic and robust representation. SIFT [19], HOG [20] and Haar-like [21] features are the representative ones. This is due to the fact that these features can produce representations associated with complex cells in human brain [19]. However, due to the diversity of appearances, illumination conditions and backgrounds, it's difficult to manually design a robust feature descriptor to perfectly describe all kinds of objects.

Classification. Besides, a classifier is needed to distinguish a target object from all the other categories and to make the representations more hierarchical, semantic and informative for visual recognition. Usually, the Supported Vector Machine (SVM) [22], AdaBoost [23] and Deformable Part-based Model
(DPM) [24] are good choices. Among these classifiers, the DPM is a flexible model by combining object parts with deformation cost to handle severe deformations. In DPM, with the aid of a graphical model, carefully designed low-level features and kinematically inspired part decompositions are combined. And discriminative learning of graphical models allows for building high-precision part-based models for a variety of object classes.

Based on these discriminant local feature descriptors and shallow learnable architectures, state of the art results have been obtained on PASCAL VOC object detection competition [25] and real-time embedded systems have been obtained with a low burden on hardware. However, small gains are obtained during 2010-2012 by only building ensemble systems and employing minor variants of successful methods [15]. This fact is due to the following reasons: 1) The generationneration of candidate bounding boxes with a sliding window strategy is redundant, inefficient and inaccurate. 2) The semantic gap cannot be

Fig. 1. The application domains of object detection.

bridged by the combination of manually engineered low-level descriptors and discriminatively-trained shallow models.

Thanks to the emergency of Deep Neural Networks (DNNs) [6][S7], a more significant gain is obtained with the introduction of Regions with CNN features (R-CNN) [15]. DNNs, or the most representative CNNs, act in a quite different way from traditional approaches. They have deeper architectures with the capacity to learn more complex features than the shallow ones. Also the expressivity and robust training algorithms allow to learn informative object representations without the need to design features manually [26].

Since the proposal of R-CNN, a great deal of improved models have been suggested, including Fast R-CNN which jointly optimizes classification and bounding box regression tasks [16], Faster R-CNN which takes an additional subnetwork to generate region proposals [18] and YOLO which accomplishes object detection via a fixed-grid regression [17]. All of them bring different degrees of detection performance improvements over the primary R-CNN and make real-time and accurate object detection become more achievable.

In this paper, a systematic review is provided to summarise representative models and their different characteristics in several application domains, including generic object detection [15], [16], [18], salient object detection [27], [28], face detection [29]–[31] and pedestrian detection [32], [33]. Their relationships are depicted in Figure 1. Based on basic CNN architectures, generic object detection is achieved with bounding box regression, while salient object detection is accomplished with local contrast enhancement and pixel-level segmentation. Face detection and pedestrian detection are closely related to generic object detection and mainly accomplished with multi-scale adaption and multi-feature fusion/boosting forest, respectively. The dotted lines indicate that the corresponding domains are associated with each other under certain conditions. It should be noticed that the covered domains are diversified. Pedestrian and face images have regular structures, while general objects and scene images have more complex variations in geometric structures and layouts. Therefore, different deep models are required by various images.

There has been a relevant pioneer effort [34] which mainly focuses on relevant software tools to implement deep learning techniques for image classification and object detection, but pays little attention on detailing specific algorithms. Different from it, our work not only reviews deep learning based object detection models and algorithms covering different application domains in detail, but also provides their corresponding experimental comparisons and meaningful analyses.

The rest of this paper is organized as follows. In Section 2, a brief introduction on the history of deep learning and the basic architecture of CNN is provided. Generic object detection architectures are presented in Section 3. Then reviews of CNN applied in several specific tasks, including salient object detection, face detection and pedestrian detection, are exhibited in Section 4-6, respectively. Several promising future directions are proposed in Section 7. At last, some concluding remarks are presented in Section 8.

II. A BRIEF OVERVIEW OF DEEP LEARNING

Prior to overview on deep learning based object detection approaches, we provide a review on the history of deep learning along with an introduction on the basic architecture and advantages of CNN.

A. The History: Birth, Decline and Prosperity

Deep models can be referred to as neural networks with deep structures. The history of neural networks can date back to 1940s [35], and the original intention was to simulate the human brain system to solve general learning problems in a principled way. It was popular in 1980s and 1990s with the proposal of back-propagation algorithm by Hinton et al. [36]. However, due to the overfitting of training, lack of large scale training data, limited computation power and insignificance in performance compared with other machine learning tools, neural networks fell out of fashion in early 2000s.

Deep learning has become popular since 2006 [37][S7] with a break through in speech recognition [38]. The recovery of deep learning can be attributed to the following factors.

•as ImageNet [39], to fully exhibit its very large learningThe emergence of large scale annotated training data, such

capacity;

•systems, such as GPU clusters;Fast development of high performance parallel computing

•and training strategies. With unsupervised and layerwiseSignificant advances in the design of network structures pre-training guided by Auto-Encoder (AE) [40] or Restricted Boltzmann Machine (RBM) [41], a good initialization is provided. With dropout and data augmentation, the overfitting problem in training has been relieved [6], [42]. With batch normalization (BN), the training of very deep neural networks becomes quite efficient [43]. Meanwhile, various network structures, such as AlexNet [6], Overfeat [44], GoogLeNet [45], VGG [46] and ResNet [47], have been extensively studied to improve the performance.

What prompts deep learning to have a huge impact on the entire academic community? It may owe to the contribution of Hinton's group, whose continuous efforts have demonstrated that deep learning would bring a revolutionary breakthrough on grand challenges rather than just obvious improvements on small datasets. Their success results from training a large CNN on 1.2 million labeled images together with a few techniques [6] (e.g., ReLU operation [48] and 'dropout' regularization).

B. Architecture and Advantages of CNN

CNN is the most representative model of deep learning [26]. A typical CNN architecture, which is referred to as VGG16, can be found in Fig. S1. Each layer of CNN is known as a feature map. The feature map of the input layer is a 3D matrix of pixel intensities for different color channels (e.g. RGB). The feature map of any internal layer is an induced multi-channel image, whose 'pixel' can be viewed as a specific feature. Every neuron is connected with a small portion of adjacent neurons from the previous layer (receptive field). Different types of transformations [6], [49], [50] can be conducted on feature maps, such as filtering and pooling. Filtering (convolution) operation convolutes a filter matrix (learned weights) with the values of a receptive field of neurons and takes a nonlinear function (such as sigmoid [51], ReLU) to obtain final responses. Pooling operation, such as max pooling, average pooling, L2-pooling and local contrast normalization [52], summaries the responses of a receptive field into one value to produce more robust feature descriptions.

With an interleave between convolution and pooling, an initial feature hierarchy is constructed, which can be fine-tuned in a supervised manner by adding several fully connected (FC) layers to adapt to different visual tasks. According to the tasks involved, the final layer with different activation functions [6] is added to get a specific conditional probability for each output neuron. And the whole network can be optimized on an objective function (e.g. mean squared error or cross-entropy loss) via the stochastic gradient descent (SGD) method. The typical VGG16 has totally 13 convolutional (conv) layers, 3 fully connected layers, 3 max-pooling layers and a softmax classification layer. The conv feature maps are produced by convoluting 3*3 filter windows, and feature map resolutions are reduced with 2 stride max-pooling layers. An arbitrary test image of the same size as training samples can be processed with the trained network. Re-scaling or cropping operations may be needed if different sizes are provided [6].

The advantages of CNN against traditional methods can be summarised as follows.

•level representations from pixel to high-level semantic fea-Hierarchical feature representation, which is the multitures learned by a hierarchical multi-stage structure [15], [53], can be learned from data automatically and hidden factors of input data can be disentangled through multi-level nonlinear mappings.

•architecture provides an exponentially increased expressiveCompared with traditional shallow models, a deeper capability.

• The architecture of CNN provides an opportunity to jointly optimize several related tasks together (e.g. Fast RCNN combines classification and bounding box regression into a multi-task leaning manner).

•CNNs, some classical computer vision challenges can beBenefitting from the large learning capacity of deep recast as high-dimensional data transform problems and solved from a different viewpoint.

Due to these advantages, CNN has been widely applied into many research fields, such as image super-resolution reconstruction [54], [55], image classification [5], [56], image retrieval [57], [58], face recognition [8][S5], pedestrian detection [59]–[61] and video analysis [62], [63].

III. GENERIC OBJECT DETECTION

Generic object detection aims at locating and classifying existing objects in any one image, and labeling them with rectangular bounding boxes to show the confidences of existence. The frameworks of generic object detection methods can mainly be categorized into two types (see Figure 2). One follows traditional object detection pipeline, generating region proposals at first and then classifying each proposal into different object categories. The other regards object detection as a regression or classification problem, adopting a unified framework to achieve final results (categories and locations) directly. The region proposal based methods mainly include R-CNN [15], SPP-net [64], Fast R-CNN [16], Faster R-CNN [18], R-FCN [65], FPN [66] and Mask R-CNN [67], some of which are correlated with each other (e.g. SPP-net modifies RCNN with a SPP layer). The regression/classification based methods mainly includes MultiBox [68], AttentionNet [69], G-CNN [70], YOLO [17], SSD [71], YOLOv2 [72], DSSD [73] and DSOD [74]. The correlations between these two pipelines are bridged by the anchors introduced in Faster RCNN. Details of these methods are as follows.

A. Region Proposal Based Framework

The region proposal based framework, a two-step process, matches the attentional mechanism of human brain to some extent, which gives a coarse scan of the whole scenario firstly and then focuses on regions of interest. Among the pre-related works [44], [75], [76], the most representative one is Overfeat [44]. This model inserts CNN into sliding window method, which predicts bounding boxes directly from locations of the topmost feature map after obtaining the confidences of underlying object categories.

1) R-CNN: It is of significance to improve the quality of candidate bounding boxes and to take a deep architecture to extract high-level features. To solve these problems, R-CNN [15] was proposed by Ross Girshick in 2014 and obtained a mean average precision (mAP) of 53.3% with more than 30% improvement over the previous best result (DPM HSC [77]) on PASCAL VOC 2012. Figure 3 shows the flowchart of R-CNN, which can be divided into three stages as follows.
Region proposal generation. The R-CNN adopts selective search [78] to generate about 2k region proposals for each image. The selective search method relies on simple bottom-up grouping and saliency cues to provide more accurate candidate boxes of arbitrary sizes quickly and to reduce the searching space in object detection [24], [39].
CNN based deep feature extraction. In this stage, each region proposal is warped or cropped into a fixed resolution and the CNN module in [6] is utilized to extract a 4096dimensional feature as the final representation. Due to large learning capacity, dominant expressive power and hierarchical structure of CNNs, a high-level, semantic and robust feature representation for each region proposal can be obtained. Classification and localization. With pre-trained categoryspecific linear SVMs for multiple classes, different region proposals are scored on a set of positive regions and background (negative) regions. The scored regions are then adjusted with

R-FCN

Fig. 2. Two types of frameworks: region proposal based and regression/classification based. SPP: Spatial Pyramid Pooling [64], FRCN: Faster R-CNN [16], RPN: Region Proposal Network [18], FCN: Fully Convolutional Network [65], BN: Batch Normalization [43], Deconv layers: Deconvolution layers [54].

R-CNN: *Regions with CNN features*

Tv monitor? no.

1. Input **2**. Extract region **3**. Compute **4**. Classify image proposals (~2k) CNN features regions

Fig. 3. The flowchart of R-CNN [15], which consists of 3 stages: (1) extracts bottom-up region proposals, (2) computes features for each proposal using a CNN, and then (3) classifies each region with class-specific linear SVMs.

bounding box regression and filtered with a greedy nonmaximum suppression (NMS) to produce final bounding boxes for preserved object locations.

When there are scarce or insufficient labeled data, pretraining is usually conducted. Instead of unsupervised pretraining [79], R-CNN firstly conducts supervised pre-training on ILSVRC, a very large auxiliary dataset, and then takes a domain-specific fine-tuning. This scheme has been adopted by most of subsequent approaches [16], [18].

In spite of its improvements over traditional methods and significance in bringing CNN into practical object detection, there are still some disadvantages.

•fixed-size (e.g.,Due to the existence of FC layers, the CNN requires a$_{227}$×227) input image, which directly leads to the re-computation of the whole CNN for each evaluated region, taking a great deal of time in the testing period.

•a convolutional network (ConvNet) on object proposals isTraining of R-CNN is a multi-stage pipeline. At first, fine-tuned. Then the softmax classifier learned by finetuning is replaced by SVMs to fit in with ConvNet features. Finally, bounding-box regressors are trained.

•extracted from different region proposals and stored on theTraining is expensive in space and time. Features are disk. It will take a long time to process a relatively small training set with very deep networks, such as VGG16. At the same time, the storage memory required by these features should also be a matter of concern.

•with relatively high recalls, the obtained region proposalsAlthough selective search can generate region proposals are still redundant and this procedure is time-consuming (around 2 seconds to extract 2k region proposals).

To solve these problems, many methods have been proposed. GOP [80] takes a much faster geodesic based segmentation to replace traditional graph cuts. MCG [81] searches different scales of the image for multiple hierarchical segmentations and combinatorially groups different regions to produce proposals. Instead of extracting visually distinct segments, the edge boxes method [82] adopts the idea that objects are more likely to exist in bounding boxes with fewer contours straggling their boundaries. Also some researches tried to re-rank or refine pre-extracted region proposals to remove unnecessary ones and obtained a limited number of valuable ones, such as DeepBox [83] and SharpMask [84].

In addition, there are some improvements to solve the problem of inaccurate localization. Zhang et al. [85] utilized a bayesian optimization based search algorithm to guide the regressions of different bounding boxes sequentially, and trained class-specific CNN classifiers with a structured loss to penalize the localization inaccuracy explicitly. Saurabh Gupta et al. improved object detection for RGB-D images with semantically rich image and depth features [86], and learned a new geocentric embedding for depth images to encode each pixel. The combination of object detectors and superpixel classification framework gains a promising result on semantic scene segmentation task. Ouyang et al. proposed a deformable deep CNN (DeepID-Net) [87] which introduces a novel deformation constrained pooling (def-pooling) layer to impose geometric penalty on the deformation of various object parts and makes an ensemble of models with different settings. Lenc et al. [88] provided an analysis on the role of proposal generation in CNN-based detectors and tried to replace this stage with a constant and trivial region generation scheme. The goal is achieved by biasing sampling to match the statistics of the ground truth bounding boxes with K-means clustering. However, more candidate boxes are required to achieve comparable results to those of R-CNN.

2) SPP-net: FC layers must take a fixed-size input. That's why R-CNN chooses to warp or crop each region proposal into the same size. However, the object may exist partly in the cropped region and unwanted geometric distortion may be produced due to the warping operation. These content losses or distortions will reduce recognition accuracy, especially when the scales of objects vary.

To solve this problem, He et al. took the theory of spatial pyramid matching (SPM) [89], [90] into consideration and proposed a novel CNN architecture named SPP-net [64]. SPM takes several finer to coarser scales to partition the image into a number of divisions and aggregates quantized local features into mid-level representations.

The architecture of SPP-net for object detection can be found in Figure 4. Different from R-CNN, SPP-net reuses feature maps of the 5-th conv layer (conv5) to project region

Fig. 5. The architecture of Fast R-CNN [16].

proposals of arbitrary sizes to fixed-length feature vectors. The feasibility of the reusability of these feature maps is due to the fact that the feature maps not only involve the strength of local responses, but also have relationships with their spatial positions [64]. The layer after the final conv layer is referred to as spatial pyramid pooling layer (SPP layer). If the number of feature maps in conv5 is 256, taking a 3-level pyramid, the final feature vector for each region proposal obtained after SPP layer has a dimension of $256 \times (1^2 + 2^2 + 4^2) = 5376$.

SPP-net not only gains better results with correct estimation of different region proposals in their corresponding scales, but also improves detection efficiency in testing period with the sharing of computation cost before SPP layer among different proposals.

6) FPN: Feature pyramids built upon image pyramids (featurized image pyramids) have been widely applied in many object detection systems to improve scale invariance [24], [64] (Figure 7(a)). However, training time and memory consumption increase rapidly. To this end, some techniques take only a single input scale to represent high-level semantics and increase the robustness to scale changes (Figure 7(b)), and image pyramids are built at test time which results in an inconsistency between train/test-time inferences [16], [18]. The in-network feature hierarchy in a deep ConvNet produces feature maps of different spatial resolutions while introduces large semantic gaps caused by different depths (Figure 7(c)). To avoid using low-level features, pioneer works [71], [95] usually build the pyramid starting from middle layers or just sum transformed feature responses, missing the higher-

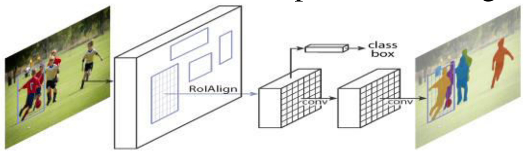

Fig. 8. The Mask R-CNN framework for instance segmentation [67].

resolution maps of the feature hierarchy.

understanding, we should not only concentrate on classifying different images, but also try to precisely estimate the concepts and locations of objects contained in each image. This task is referred as object detection [1][S1], which usually consists of different subtasks such as face detection [2][S2], pedestrian detection [3][S2] and skeleton detection [4][S3]. As one of the fundamental computer vision problems, object detection is able to provide valuable information for semantic understanding of images and videos, and is related to many applications, including image classification [5], [6], human behavior analysis [7][S4], face recognition [8][S5] and autonomous driving [9], [10]. Meanwhile, Inheriting from neural networks and related learning systems, the progress in these fields will develop neural network algorithms, and will also have great impacts on object detection techniques which can be considered as learning systems. [11]–[14][S6]. However, due to large variations in viewpoints, poses, occlusions and lighting conditions, it's difficult to perfectly accomplish object detection with an additional

Zhong-Qiu Zhao, Peng Zheng and Shou-Tao Xu are with the College of Computer Science and Information Engineering, Hefei University of Technology, China. Xindong Wu is with the School of Computing and Informatics, University of Louisiana at Lafayette, USA.

Manuscript received August xx, 2017; revised xx xx, 2017.

object localization task. So much attention has been attracted to this field in recent years [15]–[18].

The problem definition of object detection is to determine where objects are located in the given image (object localization) and which category each object belongs to (object classification). So the pipeline of traditional object detection models can be mainly divided into three stages: informative region selection, feature extraction and classification.

Informative region selection. As different objects may appear in any positions of the image and have different aspect ratios or sizes, it is a natural choice to scan the whole image with a multi-scale sliding window. Although this exhaustive strategy can find out all possible positions of the objects, its shortcomings are also obvious. Due to a large number of candidate windows, it is computationally expensive and produces too many redundant windows. However, if only a fixed number of sliding window templates are applied, unsatisfactory regions may be produced.

Feature extraction. To recognize different objects, we need to extract visual features which can provide a semantic and robust representation. SIFT [19], HOG [20] and Haar-like [21] features are the representative ones. This is due to the fact that these features can produce representations associated with complex cells in human brain [19]. However, due to the diversity of appearances, illumination conditions and backgrounds, it's difficult to manually design a robust feature descriptor to perfectly describe all kinds of objects.

Classification. Besides, a classifier is needed to distinguish a target object from all the other categories and to make the representations more hierarchical, semantic and informative for visual recognition. Usually, the Supported Vector Machine (SVM) [22], AdaBoost [23] and Deformable Part-based Model
(DPM) [24] are good choices. Among these classifiers, the DPM is a flexible model by combining object parts with deformation cost to handle severe deformations. In DPM, with the aid of a graphical model, carefully designed low-level features and kinematically inspired part decompositions are combined. And discriminative learning of graphical models allows for building high-precision part-based models for a variety of object classes.

Based on these discriminant local feature descriptors and shallow learnable architectures, state of the art results have been obtained on PASCAL VOC object detection competition [25] and real-time embedded systems have been obtained with a low burden on hardware. However, small gains are obtained during 2010-2012 by only building ensemble systems and employing minor variants of successful methods [15]. This fact is due to the following reasons: 1) The generation*B. Regression/Classification Based Framework*

Region proposal based frameworks are composed of several correlated stages, including region proposal generation, feature extraction with CNN, classification and bounding box regression, which are usually trained separately. Even in recent end-to-end module Faster R-CNN, an alternative training is still required to obtain shared convolution parameters between RPN and detection network. As a result, the time spent in handling different components becomes the bottleneck in realtime application.

One-step frameworks based on global regression/classification, mapping straightly from image pixels to bounding box coordinates and class probabilities, can reduce time expense. We firstly reviews some pioneer

CNN models, and then focus on two significant frameworks, namely You only look once (YOLO) [17] and Single Shot MultiBox Detector (SSD) [71].

1) Pioneer Works: Previous to YOLO and SSD, many researchers have already tried to model object detection as a regression or classification task.

Szegedy et al. formulated object detection task as a DNNbased regression [118], generating a binary mask for the test image and extracting detections with a simple bounding box inference. However, the model has difficulty in handling overlapping objects, and bounding boxes generated by direct upsampling is far from perfect.

Pinheiro et al. proposed a CNN model with two branches: one generates class agnostic segmentation masks and the other predicts the likelihood of a given patch centered on an object [119]. Inference is efficient since class scores and segmentation can be obtained in a single model with most of the CNN operations shared.

Erhan et al. proposed regression based MultiBox to produce scored class-agnostic region proposals [68], [120]. A unified loss was introduced to bias both localization and confidences of multiple components to predict the coordinates of classagnostic bounding boxes. However, a large quantity of additional parameters are introduced to the final layer.

Yoo et al. adopted an iterative classification approach to handle object detection and proposed an impressive end-toend CNN architecture named AttentionNet [69]. Starting from the top-left (TL) and bottom-right (BR) corner of an image, AttentionNet points to a target object by generating quantized weak directions and converges to an accurate object boundary box with an ensemble of iterative predictions. However, the model becomes quite inefficient when handling multiple categories with a progressive two-step procedure.

Najibi et al. proposed a proposal-free iterative grid based object detector (G-CNN), which models object detection as

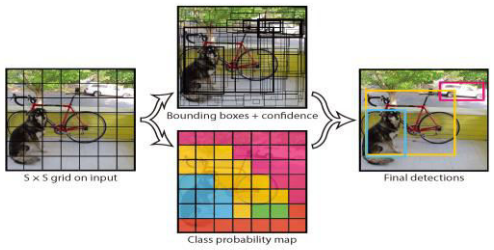

Fig. 9. Main idea of YOLO [17].

finding a path from a fixed grid to boxes tightly surrounding the objects [70]. Starting with a fixed multi-scale bounding box grid, G-CNN trains a regressor to move and scale elements of the grid towards objects iteratively. However, G-CNN has a difficulty in dealing with small or highly overlapping objects.

2) YOLO: Redmon et al. [17] proposed a novel framework called YOLO, which makes use of the whole topmost feature map to predict both confidences for multiple categories and bounding boxes. The basic idea of YOLO is exhibited in Figure 9. YOLO divides the input image into an S × S grid and each grid cell is responsible for predicting the object centered in that grid cell. Each grid cell predicts *B* bounding boxes and their corresponding confidence scores. Formally, confidence scores are defined as $Pr(Object) * IOU^{truth}_{pred}$, which shows confidences of its prediction (indicates how likely there exist objects (IOU^{truth}_{pred} $Pr(Object)$. At the same) ≥ 0) and time, regardless of the number of boxes, *C* conditional class probabilities ($Pr(Class^i|Object)$) should also be predicted in each grid cell. It should be noticed that only the contribution from the grid cell containing an object is calculated.

At test time, class-specific confidence scores for each box are achieved by multiplying the individual box confidence predictions with the conditional class probabilities as follows.

$$\begin{aligned} &Pr(Object) * IOU_{pred}^{truth} * Pr(Class_i|Object) \\ &\quad = Pr(Class_i) * IOU_{pred}^{truth} \end{aligned} \qquad (5)$$

where the existing probability of class-specific objects in the box and the fitness between the predicted box and the object are both taken into consideration.

During training, the following loss function is optimized,

$$\begin{aligned} &\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\right] \\ &+\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2\right] \\ &+\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left(C_i - \hat{C}_i\right)^2 \\ &+\lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} \left(C_i - \hat{C}_i\right)^2 \\ &+\sum_{i=0}^{S^2} \mathbb{1}_{i}^{obj} \sum_{c\in classes} (p_i(c) - \hat{p}_i(c))^2 \end{aligned} \qquad (6)$$

In a certain cell i, (x_i, y_i) denote the center of the box relative to the bounds of the grid cell, (w_i, h_i) are the normalized width and height relative to the image size, C_i represents confidence scores, $1^{obj}{}_i$ indicates the existence of objects and $1^{obj}{}_{ij}$ denotes that the prediction is conducted by the jth bounding box predictor. Note that only when an object is present in that grid cell, the loss function penalizes classification errors. Similarly, when the predictor is 'responsible' for the ground truth box (i.e. the highest IoU of any predictor in that grid cell is achieved), bounding box coordinate errors are penalized.

The YOLO consists of 24 conv layers and 2 FC layers, of which some conv layers construct ensembles of inception modules with 1 × 1 reduction layers followed by 3 × 3 conv layers. The network can process images in real-time at 45 FPS and a simplified version Fast YOLO can reach 155 FPS with better results than other real-time detectors. Furthermore, YOLO produces fewer false positives on background, which makes the cooperation with Fast R-CNN become possible. An improved version, YOLOv2, was later proposed in [72], which adopts several impressive strategies, such as BN, anchor boxes, dimension cluster and multi-scale training.

3) SSD: YOLO has a difficulty in dealing with small objects in groups, which is caused by strong spatial constraints imposed on bounding box predictions [17]. Meanwhile, YOLO struggles to generalize to objects in new/unusual aspect ratios/ configurations and produces relatively coarse features due to multiple downsampling operations.

Aiming at these problems, Liu et al. proposed a Single Shot MultiBox Detector (SSD) [71], which was inspired by the anchors adopted in MultiBox [68], RPN [18] and multi-scale representation [95]. Given a specific feature map, instead of fixed grids adopted in YOLO, the SSD takes advantage of a set of default anchor boxes with different aspect ratios and scales to discretize the output space of bounding boxes. To handle objects with various sizes, the network fuses predictions from multiple feature maps with different resolutions .

The architecture of SSD is demonstrated in Figure 10. Given the VGG16 backbone architecture, SSD adds several feature layers to the end of the network, which are responsible for predicting the offsets to default boxes with different scales and aspect ratios and their associated confidences. The network is trained with a weighted sum of localization loss (e.g. Smooth L1) and confidence loss (e.g. Softmax), which is similar to (1). Final detection results are obtained by conducting NMS on multi-scale refined bounding boxes.

Integrating with hard negative mining, data augmentation and a larger number of carefully chosen default anchors, SSD significantly outperforms the Faster R-CNN in terms of accuracy on PASCAL VOC and COCO, while being three times faster. The SSD300 (input image size is 300×300) runs at 59 FPS, which is more accurate and efficient than YOLO. However, SSD is not skilled at dealing with small objects, which can be relieved by adopting better feature extractor backbone (e.g. ResNet101), adding deconvolution layers with skip connections to introduce additional large-scale context [73] and designing better network structure (e.g. Stem Block and Dense Block) [74].

Fig. 10. The architecture of SSD 300 [71]. SSD adds several feature layers to the end of VGG16 backbone network to predict the offsets to default anchor boxes and their associated confidences. Final detection results are obtained by conducting NMS on multi-scale refined bounding boxes.

C. Experimental Evaluation

We compare various object detection methods on three benchmark datasets, including PASCAL VOC 2007 [25], PASCAL VOC 2012 [121] and Microsoft COCO [94]. The evaluated approaches include R-CNN [15], SPP-net [64], Fast R-CNN [16], NOC [114], Bayes [85], MR-CNN&S-CNN
[105], Faster R-CNN [18], HyperNet [101], ION [95], MS-
GR [104], StuffNet [100], SSD300 [71], SSD512 [71], OHEM
[113], SDP+CRC [33], GCNN [70], SubCNN [60], GBD-Net
[109], PVANET [116], YOLO [17], YOLOv2 [72], R-FCN
[65], FPN [66], Mask R-CNN [67], DSSD [73] and DSOD [74]. If no specific instructions for the adopted framework are provided, the utilized model is a VGG16 [46] pretrained on 1000-way ImageNet classification task [39]. Due to the limitation of paper length, we only provide an overview, including proposal, learning method, loss function, programming language and platform, of the prominent architectures in Table I. Detailed experimental settings, which can be found in the original papers, are missed. In addition to the comparisons of detection accuracy, another comparison is provided to evaluate their test consumption on PASCAL VOC 2007.

1) PASCAL VOC 2007/2012: PASCAL VOC 2007 and
2012 datasets consist of 20 categories. The evaluation terms are Average Precision (AP) in each single category and mean Average Precision (mAP) across all the 20 categories. Comparative results are exhibited in Table II and III, from which the following remarks can be obtained.

•bone CNN models can definitely improve object detectionIf incorporated with a proper way, more powerful backperformance (the comparison among R-CNN with AlexNet, R-CNN with VGG16 and SPP-net with ZF-Net [122]).

•end multi-task architecture (FRCN) and RPN (Faster R-With the introduction of SPP layer (SPP-net), end-to-

CNN), object detection performance is improved gradually and apparently.

•obtain multi-level robust features, data augmentation is veryDue to large quantities of trainable parameters, in order to important for deep learning based models (Faster R-CNN with '07' ,'07+12' and '07+12+coco').

•affecting object detection performance, such as multi-scaleApart from basic models, there are still many other factors and multi-region feature extraction (e.g. MR-CNN), modified classification networks (e.g. NOC), additional information from other correlated tasks (e.g. StuffNet, HyperNet), multi-scale representation (e.g. ION) and mining of hard negative samples (e.g. OHEM).

• As YOLO is not skilled in producing object localizations
of high IoU, it obtains a very poor result on VOC 2012. However, with the complementary information from Fast R-CNN (YOLO+FRCN) and the aid of other strategies, such as anchor boxes, BN and fine grained features, the localization errors are corrected (YOLOv2).

•network as a fully convolutional one, R-FCN achieves aBy combining many recent tricks and modelling the whole more obvious improvement of detection performance over other approaches.

2) Microsoft COCO: Microsoft COCO is composed of 300,000 fully segmented images, in which each image has an average of 7 object instances from a total of 80 categories. As there are a lot of less iconic objects with a broad range of scales and a stricter requirement on object localization, this dataset is more challenging than PASCAL 2012. Object detection performance is evaluated by AP computed under different degrees of IoUs and on different object sizes. The results are shown in Table IV.

Besides similar remarks to those of PASCAL VOC, some other conclusions can be drawn as follows from Table IV.

•ing object detection performance, which provide additionalMulti-scale training and test are beneficial in improvinformation in different resolutions (R-FCN). FPN and DSSD provide some better ways to build feature pyramids to achieve multi-scale representation. The complementary information from other related tasks is also helpful for accurate object localization (Mask R-CNN with instance segmentation task).

•Faster R-CNN and R-FCN, perform better than regres-Overall, region proposal based methods, such as sion/classfication based approaches, namely YOLO and SSD, due to the fact that quite a lot of localization errors are produced by regression/classfication based approaches.

•which provides additional information by consulting nearbyContext modelling is helpful to locate small objects, objects and surroundings (GBD-Net and multi-path).

•small objects, the results on this dataset are much worseDue to the existence of a large number of nonstandard than those of VOC 2007/2012. With the introduction of other powerful frameworks (e.g. ResNeXt [123]) and useful strategies (e.g. multi-task learning [67], [124]), the performance can be improved.

•importance of network design to release the requirementsThe success of DSOD in training from scratch stresses the for perfect pre-trained classifiers on relevant tasks and large numbers of annotated samples.

3) Timing Analysis: Timing analysis (Table V) is conducted on Intel i7-6700K CPU with a single core and NVIDIA Titan

TABLE I
AN OVERVIEW OF PROMINENT GENERIC OBJECT DETECTION ARCHITECTURES.

Framework	Proposal	Multi-scale Input	Learning Method	Loss Function	Softmax Layer	End-to-end Train	Platform	Language
R-CNN [15]	Selective Search	-	SGD,BP	Hinge loss (classification),Bounding box regression	+	-	Caffe	Matlab
SPP-net [64]	EdgeBoxes	+	SGD	Hinge loss (classification),Bounding box regression	+	-	Caffe	Matlab
Fast RCNN [16]	Selective Search	+	SGD	Class Log loss+bounding box regression	+	-	Caffe	Python
Faster R-CNN [18]	RPN	+	SGD	Class Log loss+bounding box regression	+	+	Caffe	Python/Matlab
R-FCN [65]	RPN	+	SGD	Class Log loss+bounding box regression	-	+	Caffe	Matlab
Mask R-CNN [67]	RPN	+	SGD	Class Log loss+bounding box regression+Semantic sigmoid loss	+	+	TensorFlow/Keras	Python
FPN [66]	RPN	+	Synchronized SGD	Class Log loss+bounding box regression	+	+	TensorFlow	Python
YOLO [17]	-	-	SGD	Class sum-squared error loss+bounding box regression+object confidence+background confidence	+	+	Darknet	C
SSD [71]	-	-	SGD	Class softmax loss+bounding box regression	-	+	Caffe	C++
YOLOv2 [72]	-	-	SGD	Class sum-squared error loss+bounding box regression+object confidence+background confidence	+	+	Darknet	C

[*] '+' denotes that corresponding techniques are employed while '-' denotes that this technique is not considered. It should be noticed that R-CNN and SPP-net can not be trained end-to-end with a multi-task loss while the other architectures are based on multi-task joint training. As most of these architectures are re-implemented on different platforms with various programming languages, we only list the information associated with the versions

by the referenced authors.

TABLE II
COMPARATIVE RESULTS ON VOC 2007 TEST SET (%).

Methods	Trained on	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN (Alex) [15]	07	68.1	72.8	56.8	43.0	36.8	66.3	74.2	67.6	34.4	63.5	54.5	61.2	69.1	68.6	58.7	33.4	62.9	51.1	62.5	68.6	58.5

R-CNN(VGG16) [15] 0773.477.063.445.4 44.675.178.1 79.840.573.762.279.478.1 73.164.235.6
66.8 67.2 70.471.166.0
SPP-net(ZF) [64] 0768.571.758.7 41.942.567.7 72.173.834.7 67.063.466.072.571.3 58.932.8
60.9 56.1 67.968.860.9
GCNN [70] 07 68.377.368.5 52.438.678.5 79.581.047.1 73.664.577.280.575.866.6 34.365.2
64.4 75.6 66.466.8
Bayes [85] 07 74.183.267.0 50.851.676.2 81.477.248.1 78.965.677.378.475.170.1 41.469.6
60.8 70.2 73.768.5
Fast R-CNN [16] 07+1277.0 78.169.359.438.3 81.678.686.7 42.878.868.984.782.076.6 69.931.8
70.1 74.8 80.470.470.0
SDP+CRC [33]07 76.179.468.2 52.646.078.4 78.481.046.7 73.565.378.681.076.777.3 39.065.1
67.2 77.5 70.368.9
SubCNN [60] 07 70.280.569.5 60.347.979.0 78.784.248.5 73.963.082.780.676.070.2 38.262.4
67.7 77.7 60.568.5
StuffNet30 [100] 0772.681.770.6 60.553.081.5 83.783.952.2 78.970.785.085.777.0 78.742.2
73.6 69.2 79.273.872.7
NOC [114] 07+1276.381.474.4 61.760.884.7 78.282.953.0 79.269.283.283.278.568.0 45.071.6
76.7 82.2 75.773.3
MR-CNN&S-CNN [110]07+1280.3 84.178.570.8 68.588.085.9 87.860.385.273.787.2 86.585.0
76.4 48.576.375.5 85.081.078.2
HyperNet [101]07+12 77.483.375.0 69.162.483.1 87.487.457.1 79.871.485.185.180.0 79.151.2
79.1 75.7 80.976.576.3
MS-GR [104]07+1280.081.0 77.472.164.388.2 88.188.464.4 85.473.187.387.485.179.6 50.178.4
79.5 86.9 75.578.6
OHEM+Fast R-CNN [113]07+1280.685.779.8 69.960.888.3 87.989.659.785.176.587.1 87.382.4
78.8 53.780.578.7 84.580.778.9
ION [95] 07+12+S 80.2 85.2 78.8 70.9 62.6 86.6 86.9 89.8 61.7 86.9
76.5 88.4 87.5 83.4 80.5 52.4 78.1 77.2 86.9 83.5 79.2 Faster R-CNN [18]
07 70.0 80.6 70.1 57.3 49.9 78.2 80.4 82.0 52.2 75.3 67.2 80.3
79.8 75.0 76.3 39.1 68.3 67.3 81.1 67.6 69.9 Faster R-CNN [18] 07+12
76.5 79.0 70.9 65.5 52.1 83.1 84.7 86.4 52.0 81.9 65.7 84.8 84.6
77.5 76.7 38.8 73.6 73.9 83.0 72.6 73.2
Faster R-CNN [18] 07+12+COCO 84.3 82.0 77.7 68.9 65.7 88.1 88.4 88.9 63.6
86.3 70.8 85.9 87.6 80.1 82.3 53.6 80.4 75.8 86.6 78.9 78.8 SSD300 [71]
07+12+COCO 80.9 86.3 79.0 76.2 57.6 87.3 88.2 88.6 60.5 85.4 76.7
87.5 89.2 84.5 81.4 55.0 81.9 81.5 85.9 78.9 79.6
SSD512 [71]07+12+COCO86.688.3 82.476.066.3 88.688.989.1 65.188.473.686.588.9 85.384.6
59.1 85.0 80.487.481.2 81.6

* '07': VOC2007 trainval, '07+12': union of VOC2007 and VOC2012 trainval, '07+12+COCO': trained on COCO trainval35k at first and then fine-tuned on 07+12. The S in ION '07+12+S' denotes SBD segmentation labels.

TABLE III
COMPARATIVE RESULTS ON VOC 2012 TEST SET (%).

Methods Trained onareo bikebirdboatbottlebus car catchaircowtable doghorsembikepersonplant
sheep sofa traintvmAP

R-CNN(Alex) [15] 1271.865.852.0 34.132.659.6 60.069.827.6 52.041.769.661.368.3 57.829.6
57.8 40.9 59.354.153.3
R-CNN(VGG16) [15] 1279.672.761.941.2 41.965.966.4 84.638.567.246.782.074.8 76.065.235.6
65.4 54.2 67.460.362.4
Bayes [85] 12 82.976.164.1 44.649.470.3 71.284.642.7 68.655.882.777.179.968.7 41.469.0
60.0 72.0 66.266.4
Fast R-CNN [16] 07++12 82.378.470.852.3 38.777.871.6 89.344.273.055.087.580.5 80.872.035.1
68.3 65.7 80.464.268.4
SutffNet30 [100] 1283.076.971.2 51.650.176.4 75.787.848.3 74.855.785.781.280.3 79.544.2
71.8 61.0 78.565.470.0
NOC [114] 07+1282.879.071.6 52.353.774.1 69.084.946.9 74.353.185.081.379.572.2 38.972.4
59.5 76.7 68.168.8
MR-CNN&S-CNN [110]07++12 85.582.976.6 57.862.779.4 77.286.655.079.162.287.0 83.484.7
78.9 45.373.465.8 80.374.073.9
HyperNet [101]07++1284.278.5 73.655.653.7 78.779.887.7 49.674.952.186.081.783.3 81.848.6
73.5 59.4 79.965.771.4
OHEM+Fast R-CNN [113]07++12+coco90.187.479.9 65.866.386.185.092.962.4 83.469.590.688.9
88.9 83.659.082.0 74.788.277.3 80.1
ION [95] 07+12+S 87.5 84.7 76.8 63.8 58.3 82.6 79.0 90.9 57.8 82.0 64.7 88.9 86.5 84.7 82.3 51.4 78.2
69.2 85.2 73.5 76.4 Faster R-CNN [18] 07++12 84.9 79.8 74.3 53.9 49.8 77.5 75.9 88.5 45.6 77.1 55.3
86.9 81.7 80.9 79.6 40.1 72.6 60.9 81.2 61.5 70.4
Faster R-CNN [18]07++12+coco 87.483.676.8 62.959.681.9 82.091.354.982.659.089.0 85.584.7
84.1 52.2 78.965.585.4 70.275.9
YOLO [17] 07++1277.067.2 57.738.322.768.3 55.981.436.2 60.848.577.272.371.363.5 28.952.2
54.8 73.9 50.857.9
YOLO+Fast R-CNN [17]07++1283.478.573.5 55.843.479.1 73.189.449.475.557.087.5 80.981.0
74.7 41.871.568.5 82.167.270.7
YOLOv2 [72] 07++12+coco 88.8 87.0 77.8 64.9 51.8 85.2 79.3 93.1 64.4
81.4 70.2 91.3 88.1 87.2 81.0 57.7 78.1 71.0 88.5 76.8 78.2 SSD300 [71]
07++12+coco 91.0 86.0 78.1 65.0 55.4 84.9 84.0 93.4 62.1 83.6 67.3
91.3 88.9 88.6 85.6 54.7 83.8 77.3 88.3 76.5 79.3
SSD512 [71]07++12+coco91.488.682.6 71.463.187.4 88.193.966.986.666.392.0 91.790.888.560.9
87.0 75.4 90.280.482.2
R-FCN (ResNet101) [16]07++12+coco92.3 89.986.774.7 75.286.789.095.870.290.4 66.595.093.2
92.1 91.171.089.7 76.092.083.4 85.0

* '07++12': union of VOC2007 trainval and test and VOC2012 trainval. '07++12+COCO': trained on COCO trainval35k at first then fine-tuned on 07++12.

TABLE IV
COMPARATIVE RESULTS ON MICROSOFT COCO TEST DEV SET (%).

Methods	Trained on	0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Fast R-CNN [16]	train	20.5	39.9	19.4	4.1	20.0	35.8	21.3	29.4	30.1	7.3	32.1	52.0
ION [95]	train	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
NOC+FRCN(VGG16) [114]	train	21.2	41.5	19.7	-	-	-	-	-	-	-	-	-
NOC+FRCN(Google) [114]	train	24.8	44.4	25.2	-	-	-	-	-	-	-	-	-
NOC+FRCN (ResNet101) [114]	train	27.2	48.4	27.6	-	-	-	-	-	-	-	-	-
GBD-Net [109]	train	27.0	45.8	-	-	-	-	-	-	-	-	-	-
OHEM+FRCN [113]	train	22.6	42.5	22.2	5.0	23.7	34.6	-	-	-	-	-	-
OHEM+FRCN* [113]	train	24.4	44.4	24.8	7.1	26.4	37.9	-	-	-	-	-	-
OHEM+FRCN* [113]	trainval	25.5	45.9	26.1	7.4	27.7	38.5	-	-	-	-	-	-
Faster R-CNN [18]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
YOLOv2 [72]	trainval35k	21.6	44.0	19.2	5.0	22.4	35.5	20.7	31.6	33.3	9.8	36.5	54.4
SSD300 [71]	trainval35k	23.2	41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6	37.6	56.5
SSD512 [71]	trainval35k	26.8	46.5	27.8	9.0	28.9	41.9	24.8	37.5	39.8	14.0	43.5	59.0
R-FCN (ResNet101) [65]	trainval	29.2	51.5	-	10.8	32.8	45.0	-	-	-	-	-	-
R-FCN*(ResNet101) [65]	trainval	29.9	51.9	-	10.4	32.4	43.3	-	-	-	-	-	-
R-FCN**(ResNet101) [65]	trainval	31.5	53.2	-	14.3	35.5	44.2	-	-	-	-	-	-
Multi-path [112]	trainval	33.2	51.9	36.3	13.6	37.2	47.8	29.9	46.0	48.3	23.4	56.0	66.4
FPN (ResNet101) [66]	trainval35k	36.2	59.1	39.0	18.2	39.0	48.2	-	-	-	-	-	-
Mask (ResNet101+FPN) [67]	trainval35k	38.2	60.3	41.7	20.1	41.1	50.2	-	-	-	-	-	-
Mask (ResNeXt101+FPN) [67]	trainval35k	39.8	62.3	43.4	22.1	43.2	51.2	-	-	-	-	-	-
DSSD513 (ResNet101) [73]	trainval35k	33.2	53.3	35.2	13.0	35.4	51.1	28.9	43.5	46.2	21.8	49.1	66.4
DSOD300 [74]	trainval	29.3	47.3	30.6	9.4	31.5	47.0	27.3	40.7	43.0	16.7	47.1	65.0

* FRCN*: Fast R-CNN with multi-scale training, R-FCN*: R-FCN with multi-scale training, R-FCN**: R-FCN with multi-scale training and testing, Mask: Mask R-CNN.

X GPU. Except for 'SS' which is processed with CPU, the other procedures related to CNN are all evaluated on GPU. From Table V, we can draw some conclusions as follows.

•(SPP-net), test consumption is reduced largely. Test time isBy computing CNN features on shared feature maps further reduced with the unified multi-task learning (FRCN) and removal of additional region proposal generation stage (Faster R-CNN). It's also helpful to compress the parameters of FC layers with SVD [91] (PAVNET and FRCN).

TABLE V
COMPARISON OF TESTING CONSUMPTION ON VOC 07 TEST SET.

Methods	Trained on	mAP(%)	Test time(sec/img)	Rate(FPS)

SS+R-CNN [15]	07	66.0	32.84	0.03
SS+SPP-net [64]	07	63.1	2.3	0.44
SS+FRCN [16]	07+12	66.9	1.72	0.6
SDP+CRC [33]	07	68.9	0.47	2.1
SS+HyperNet* [101]	07+12	76.3	0.20	5
MR-CNN&S-CNN [110]	07+12	78.2	30	0.03
ION [95]	07+12+S	79.2	1.92	0.5
Faster R-CNN(VGG16) [18]	07+12	73.2	0.11	9.1
Faster R-CNN(ResNet101) [18]	07+12	83.8	2.24	0.4
YOLO [17]	07+12	63.4	0.02	45
SSD300 [71]	07+12	74.3	0.02	46
SSD512 [71]	07+12	76.8	0.05	19
R-FCN(ResNet101) [65]	07+12+coco	83.6	0.17	5.9
YOLOv2(544*544) [72]	07+12	78.6	0.03	40
DSSD321(ResNet101) [73]	07+12	78.6	0.07	13.6
DSOD300 [74]	07+12+coco	81.7	0.06	17.4
PVANET+ [116]	07+12+coco	83.8	0.05	21.7
PVANET+(compress) [116]	07+12+coco	82.9	0.03	31.3

* SS: Selective Search [15], SS*: 'fast mode' Selective Search [16], HyperNet*: the speed up version of HyperNet and PAVNET+ (compresss): PAVNET with additional bounding box voting and compressed fully convolutional layers.

• It takes additional test time to extract multi-scale features and contextual information (ION and MR-RCNN&SRCNN).

•network (ResNet101 against VGG16) and this time con-It takes more time to train a more complex and deeper sumption can be reduced by adding as many layers into shared fully convolutional layers as possible (FRCN). • Regression based models can usually be processed in realtime at the cost of a drop in accuracy compared with region proposal based models. Also, region proposal based models can be modified into real-time systems with the introduction of other tricks [116] (PVANET), such as BN [43], residual connections [123].

VIII. CONCLUSION

Due to its powerful learning ability and advantages in dealing with occlusion, scale transformation and background switches, deep learning based object detection has been a research hotspot in recent years. This paper provides a detailed review on deep learning based object detection frameworks which handle different sub-problems, such as occlusion, clutter and low resolution, with different degrees of modifications on R-

CNN. The review starts on generic object detection pipelines which provide base architectures for other related tasks. Then, three other common tasks, namely salient object detection, face detection and pedestrian detection, are also briefly reviewed. Finally, we propose several promising future directions to gain a thorough understanding of the object detection landscape. This review is also meaningful for the developments in neural networks and related learning systems, which provides valuable insights and guidelines for future progress.

REFERENCES

- P. F. Felzenszwalb, R. B. Girshick, D. Mcallester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, p. 1627, 2010.
- K. K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 1, pp. 39–51, 2002.
- C. Wojek, P. Dollar, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, p. 743, 2012.
- H. Kobatake and Y. Yoshinaga, "Detection of spicules on mammogram based on skeleton analysis." *IEEE Trans. Med. Imag.*, vol. 15, no. 3, pp. 235–245, 1996.
- Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM MM*, 2014.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
- Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.
- Z. Yang and R. Nevatia, "A multi-scale cascade fully convolutional network face detector," in *ICPR*, 2016.
- C. Chen, A. Seff, A. L. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *ICCV*, 2015.
- X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *CVPR*, 2017.
- A. Dundar, J. Jin, B. Martini, and E. Culurciello, "Embedded streaming deep neural networks accelerator with applications," *IEEE Trans. Neural Netw. & Learning Syst.*, vol. 28, no. 7, pp. 1572–1583, 2017.
- R. J. Cintra, S. Duffner, C. Garcia, and A. Leite, "Low-complexity approximate convolutional neural networks," *IEEE Trans. Neural Netw. & Learning Syst.*, vol. PP, no. 99, pp. 1–12, 2018.
- S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data." *IEEE Trans. Neural Netw. & Learning Syst.*, vol. PP, no. 99, pp. 1–15, 2017.
- A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature extraction with deep neural networks by a generalized discriminant analysis." *IEEE Trans. Neural Netw. & Learning Syst.*, vol. 23, no. 4, pp. 596–608, 2012.
- R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.
- R. Girshick, "Fast r-cnn," in *ICCV*, 2015.
- J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

- S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards realtime object detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.
- D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. of Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.