Object Detection With Deep Learning: A Review

Zhong-Qiu Zhao, *Member, IEEE*, Peng Zheng, Shou-Tao Xu, and Xindong Wu, *Fellow, IEEE*

***Abstract*— Due to object detection's close relationship with video analysis and image understanding, it has attracted much research attention in recent years. Traditional object detection methods are built on handcrafted features and shallow trainable architectures. Their performance easily stagnates by constructing complex ensembles that combine multiple low-level image features with high-level context from object detectors and scene classifiers. With the rapid development in deep learning, more powerful tools, which are able to learn semantic, high-level, deeper features, are introduced to address the problems existing in traditional architectures. These models behave differently in network architecture, training strategy, and optimization function. In this paper, we provide a review of deep learning-based object detection frameworks. Our review begins with a brief introduction on the history of deep learning and its representative tool, namely, the convolutional neural network. Then, we focus on typical generic object detection architectures along with some modifications and useful tricks to improve detection performance further. As distinct specific detection tasks exhibit different characteristics, we also briefly survey several specific tasks, including salient object detection, face detection, and pedestrian detection. Experimental analyses are also provided to compare various methods and draw some meaningful conclusions. Finally, several promising directions and tasks are provided to serve as guidelines for future work in both object detection and relevant neural network-based learning systems.**

***Index Terms*— Deep learning, neural network, object detection.**

I. Introduction

TO GAIN a complete image 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 that can be considered as learning systems [11]–[14], [S6]. However, due to large variations in viewpoints, poses, occlusions, and lighting conditions, it is difficult to perfectly accomplish object detection with an additional object localization task. Therefore, 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). Therefore, the pipeline of traditional object detection models can be mainly divided into three stages: informative region selection, feature extraction, and classification.

A. 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 multiscale 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 is applied, unsatisfactory regions may be produced.

B. Feature Extraction

To recognize different objects, we need to extract visual features that can provide a semantic and robust representation. Scale-invariant feature transform [19], histograms of oriented gradients (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 is difficult to manually design a robust feature descriptor to perfectly describe all kinds of objects.

C. 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.

Manuscript received September 8, 2017; revised March 3, 2018 and July 12, 2018; accepted October 15, 2018. Date of publication January 28, 2019; date of current version October 29, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61672203, Grant 61375047, and Grant 91746209, in part by the National Key Research and Development Program of China under Grant 2016YFB1000901, and in part by the Anhui Natural Science Funds for Distinguished Young Scholar under Grant 170808J08. *(Corresponding author: Zhong-Qiu Zhao.)*

Z.-Q. Zhao, P. Zheng, and S.-T. Xu are with the College of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China (e-mail: zhongqiuzhao@gmail.com).

X. Wu is with the School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA 70504 USA.

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2018.2876865

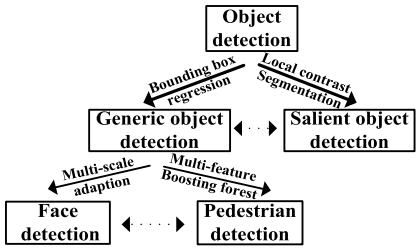


Fig. 1. Application domains of object detection.

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 visual object classes (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 of candidate bounding boxes (BBs) with a sliding window strategy is redundant, inefficient, and inaccurate and 2) the semantic gap cannot be 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], [26], [S7], a more significant gain is obtained with the introduction of regions with convolutional neural network (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 [27].

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

In this paper, a systematic review is provided to summarize representative models and their different characteristics in several application domains, including generic object detection [15]–[17], salient object detection [28], [29], face detection [30]–[32], and pedestrian detection [33], [34]. Their relationships are depicted in Fig. 1. Based on basic CNN architectures, the 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 multiscale 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 [35] 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 II, 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 III. Then, reviews of CNN applied in several specific tasks, including salient object detection, face detection, and pedestrian detection, are exhibited in Section IV–VI, respectively. Several promising future directions are proposed in Section VII. At last, some concluding remarks are presented in Section VIII.

II. Brief Overview of Deep Learning

Prior to an 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. 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 the 1940s [36], and the original intention was to simulate the human brain system to solve general learning problems in a principled way. It was popular in the 1980s and 1990s with the proposal of the back-propagation algorithm by Rumelhart *et al.* [37]. 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 the early 2000s.

Deep learning has become popular since 2006 [26], [S7], with a breakthrough in speech recognition [38]. The recovery of deep learning can be attributed to the following factors.

1) The emergence of large-scale annotated training data, such as ImageNet [39], to fully exhibit its very large learning capacity.
2) Fast development of high-performance parallel computing systems, such as GPU clusters.
3) Significant advances in the design of network structures and training strategies. With unsupervised and layerwise pretraining guided by autoencoder [40] or restricted Boltzmann machine [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 DNNs becomes quite efficient [43]. Meanwhile, various network structures, such as AlexNet [6], Overfeat [44], GoogLeNet [45], Visual Geometry Group (VGG) [46], and Residual Net (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 data sets. Their success results from training a large CNN on 1.2 million labeled images together with a few techniques [6] [e.g., rectified linear unit (ReLU) operation [48] and "dropout" regularization].

B. Architecture and Advantages of CNN

CNN is the most representative model of deep learning [27]. A typical CNN architecture, which is referred to as VGG16, can be found in Fig. S1 in the supplementary material. Each layer of CNN is known as a feature map. The feature map of the input layer is a 3-D matrix of pixel intensities for different color channels (e.g., RGB). The feature map of any internal layer is an induced multichannel 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], summarizes 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. 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 FC 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. Rescaling or cropping operations may be needed if different sizes are provided [6].

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

1) Hierarchical feature representation, which is the multilevel representations from pixel to high-level semantic features learned by a hierarchical multistage structure [15], [53], can be learned from data automatically and hidden factors of input data can be disentangled through multilevel nonlinear mappings.
2) Compared with traditional shallow models, a deeper architecture provides an exponentially increased expressive capability.
3) The architecture of CNN provides an opportunity to jointly optimize several related tasks together (e.g., fast R-CNN combines classification and bounding box regression into a multitask learning manner).
4) Benefitting from the large learning capacity of deep CNNs, some classical computer vision challenges can be recast as high-dimensional data transform problems and solved from a different viewpoint.

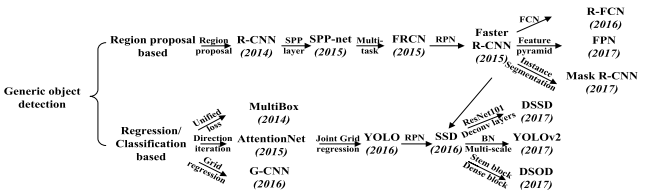


Fig. 2. Two types of frameworks: region proposal based and regression/classification based. SPP: spatial pyramid pooling [64], FRCN: faster R-CNN [16], RPN: region proposal network [17], FCN: fully convolutional network [65], BN: batch normalization [43], and Deconv layers: deconvolution layers [54]

Due to these advantages, CNN has been widely applied into many research fields, such as image superresolution 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 BBs to show the confidences of existence. The frameworks of generic object detection methods can mainly be categorized into two types (see Fig. 2). One follows the 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], spatial pyramid pooling (SPP)-net [64], Fast R-CNN [16], Faster R-CNN [17], region-based fully convolutional network (R-FCN) [65], feature pyramid networks (FPN) [66], and Mask R-CNN [67], some of which are correlated with each other (e.g., SPP-net modifies R-CNN with an SPP layer). The regression/classification-based methods mainly include MultiBox [68], AttentionNet [69], G-CNN [70], YOLO [18], Single Shot MultiBox Detector (SSD) [71], YOLOv2 [72], deconvolutional single shot detector (DSSD) [73], and deeply supervised object detectors (DSOD) [74]. The correlations between these two pipelines are bridged by the anchors introduced in Faster R-CNN. Details of these methods are as follows.

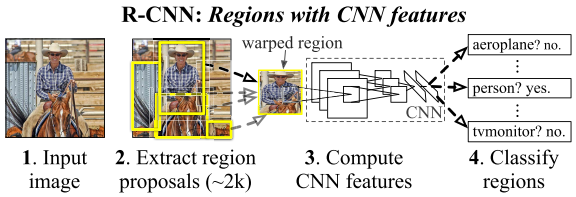


Fig. 3. Flowchart of R-CNN [15], which consists of three stages: 1) extracts BU region proposals, 2) computes features for each proposal using a CNN, and then 3) classifies each region with class-specific linear SVMs.

A. Region Proposal-Based Framework

The region proposal-based framework, a two-step process, matches the attentional mechanism of the human brain to some extent, which gives a coarse scan of the whole scenario first and then focuses on regions of interest (RoIs). Among the prerelated works [44], [75], [76], the most representative one is Overfeat [44]. This model inserts CNN into the sliding window method, which predicts BBs 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 BBs and to take a deep architecture to extract high-level features. To solve these problems, R-CNN was proposed by Girshick *et al.* [15] and obtained a mean average precision (mAP) of 53.3% with more than 30% improvement over the previous best result (DPM histograms of sparse codes [77]) on PASCAL VOC 2012. Fig. 3 shows the flowchart of R-CNN, which can be divided into three stages as follows.

a) Region Proposal Generation: The R-CNN adopts selective search [78] to generate about 2000 region proposals for each image. The selective search method relies on simple bottom-up (BU) 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].

b) 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 4096-dimensional 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.

c) Classification and Localization: With pretrained category-specific 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 bounding box regression and filtered with a greedy nonmaximum suppression (NMS) to produce final BBs for preserved object locations.

When there are scarce or insufficient labeled data, pretraining is usually conducted. Instead of unsupervised pretraining [79], R-CNN first conducts supervised pretraining on ImageNet Large-Scale Visual Recognition Competition, a very large auxiliary data set, and then takes a domain-specific fine-tuning. This scheme has been adopted by most of the subsequent approaches [16], [17].

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

1) Due to the existence of FC layers, the CNN requires a fixed size (e.g., 227×227) input image, which directly leads to the recomputation of the whole CNN for each evaluated region, taking a great deal of time in the testing period.
2) Training of R-CNN is a multistage pipeline. At first, a convolutional network (ConvNet) on object proposals is fine-tuned. Then, the softmax classifier learned by fine-tuning is replaced by SVMs to fit in with ConvNet features. Finally, bounding-box regressors are trained.
3) Training is expensive in space and time. Features are extracted from different region proposals and stored on the 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.
4) Although selective search can generate region proposals with relatively high recalls, the obtained region proposals are still redundant and this procedure is time-consuming (around 2 s to extract 2000 region proposals).

To solve these problems, many methods have been proposed. Geodesic object proposals [80] takes a much faster geodesic-based segmentation to replace traditional graph cuts. Mutiscale combinatorial grouping [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 BBs with fewer contours straggling their boundaries. Also, some studies tried to rerank or refine preextracted 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 BBs sequentially and trained class-specific CNN classifiers with a structured loss to penalize the localization inaccuracy explicitly. Gupta *et al.* [86] improved object detection for RGB-D images with semantically rich image and depth features 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 the semantic scene segmentation task. Ouyang *et al.* [87] proposed a deformable deep CNN (DeepID-Net) that 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 and Vedaldi [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 BBs with K-means clustering.

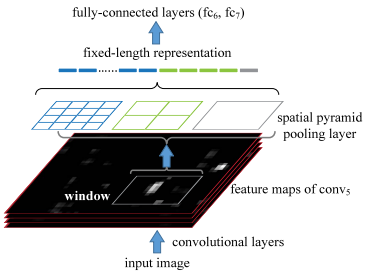


Fig. 4. Architecture of SPP-net for object detection [64].

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 is 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.* [64] took the theory of spatial pyramid matching (SPM) [89], [90] into consideration and proposed a novel CNN architecture named SPP-net. 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 Fig. 4. Different from R-CNN, SPP-net reuses feature maps of the fifth conv layer (conv5) to project region 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 the SPP layer. If the number of feature maps in conv5 is 256, taking a three-level pyramid, the final feature vector for each region proposal obtained after the SPP layer has a dimension of $256 \times (1^2 + 2^2 + 4^2) = 5376$.

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

3) Fast R-CNN: Although SPP-net has achieved impressive improvements in both accuracy and efficiency over R-CNN, it still has some notable drawbacks. SPP-net takes almost the same multistage pipeline as R-CNN, including feature extraction, network fine-tuning, SVM training, and bounding-box regressor fitting. Therefore, an additional expense on storage space is still required. In addition, the conv layers preceding the SPP layer cannot be updated with the fine-tuning algorithm introduced in [64]. As a result, an accuracy drop of very deep networks is unsurprising. To this end, Girshick [16] introduced a multitask loss on classification and bounding box regression and proposed a novel CNN architecture named Fast R-CNN.

The architecture of Fast R-CNN is exhibited in Fig. 5. Similar to SPP-net, the whole image is processed with conv layers to produce feature maps. Then, a fixed-length feature vector is extracted from each region proposal with an RoI

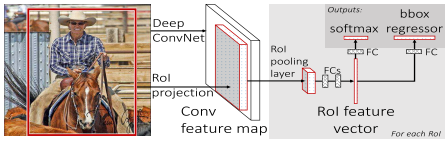


Fig. 5. Architecture of Fast R-CNN [16].

pooling layer. The RoI pooling layer is a special case of the SPP layer, which has only one pyramid level. Each feature vector is then fed into a sequence of FC layers before finally branching into two sibling output layers. One output layer is responsible for producing softmax probabilities for all $C + 1$ categories (C object classes plus one "background" class) and the other output layer encodes refined bounding-box positions with four real-valued numbers. All parameters in these procedures (except the generation of region proposals) are optimized via a multitask loss in an end-to-end way.

The multitasks loss L is defined in the following to jointly train classification and bounding-box regression:

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1]L_{\text{loc}}(t^u, v) \quad (1)$$

where $L_{\text{cls}}(p, u) = -\log p_u$ calculates the log loss for ground truth class u, and p_u is driven from the discrete probability distribution $p = (p_0, \cdots, p_C)$ over the $C+1$ outputs from the last FC layer. $L_{\text{loc}}(t^u, v)$ is defined over the predicted offsets $t^u = (t^u_x, t^u_y, t^u_w, t^u_h)$ and ground-truth bounding-box regression targets $v = (v_x, v_y, v_w, v_h)$, where x, y, w, and h denote the two coordinates of the box center, width, and height, respectively. Each t^u adopts the parameter settings in [15] to specify an object proposal with a log-space height/width shift and scale-invariant translation. The Iverson bracket indicator function $[u \geq 1]$ is employed to omit all background RoIs. To provide more robustness against outliers and eliminate the sensitivity in exploding gradients, a smooth L_1 loss is adopted to fit bounding-box regressors as follows:

$$L_{\text{loc}}(t^u, v) = \sum_{i \in x,y,w,h} \text{smooth}_{L_1}\left(t^u_i - v_i\right) \quad (2)$$

where

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise.} \end{cases} \quad (3)$$

To accelerate the pipeline of Fast R-CNN, another two tricks are of necessity. On the one hand, if training samples (i.e., RoIs) come from different images, backpropagation through the SPP layer becomes highly inefficient. Fast R-CNN samples minibatches hierarchically, namely, N images sampled randomly at first and then R/N RoIs sampled in each image, where R represents the number of RoIs. Critically, computation and memory are shared by RoIs from the same image in the forward and backward pass. On the other hand, much time is spent in computing the FC layers during the forward pass [16]. The truncated singular value decomposition (SVD) [91] can be utilized to compress large FC layers and to accelerate the testing procedure.

In the Fast R-CNN, regardless of region proposal generation, the training of all network layers can be processed in a single stage with a multitask loss. It saves the additional

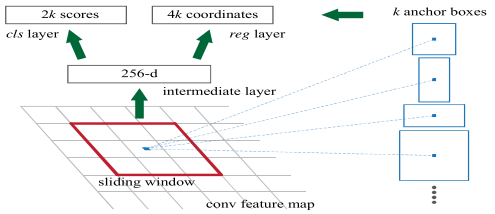


Fig. 6. RPN in Faster R-CNN [17]. K predefined anchor boxes are convoluted with each sliding window to produce fixed-length vectors which are taken by cls and reg layer to obtain corresponding outputs.

expense on storage space and improves both accuracy and efficiency with more reasonable training schemes.

4) Faster R-CNN: Despite the attempt to generate candidate boxes with biased sampling [88], state-of-the-art object detection networks mainly rely on additional methods, such as selective search and Edgebox, to generate a candidate pool of isolated region proposals. Region proposal computation is also a bottleneck in improving efficiency. To solve this problem, Ren *et al.* [17], [92] introduced an additional region proposal network (RPN), which acts in a nearly cost-free way by sharing full-image conv features with detection network.

RPN is achieved with an FCN, which has the ability to predict object bounds and scores at each position simultaneously. Similar to [78], RPN takes an image of arbitrary size to generate a set of rectangular object proposals. RPN operates on a specific conv layer with the preceding layers shared with the object detection network.

The architecture of RPN is shown in Fig. 6. The network slides over the conv feature map and fully connects to an $n \times n$ spatial window. A low-dimensional vector (512-dimensional for VGG16) is obtained in each sliding window and fed into two sibling FC layers, namely, box-classification layer (cls) and box-regression layer (reg). This architecture is implemented with an $n \times n$ conv layer followed by two sibling 1×1 conv layers. To increase nonlinearity, ReLU is applied to the output of the $n \times n$ conv layer.

The regressions toward true BBs are achieved by comparing proposals relative to reference boxes (anchors). In the Faster R-CNN, anchors of three scales and three aspect ratios are adopted. The loss function is similar to (1)

$$L(p_i, t_i) = \frac{1}{N_{\text{cls}}} \sum_i L_{\text{cls}}\left(p_i, p_i^*\right) + \lambda \frac{1}{N_{\text{reg}}} \sum_i p_i^* L_{\text{reg}}\left(t_i, t_i^*\right) \tag{4}$$

where p_i is the predicted probability of the ith anchor being an object. The ground truth label p_i^* is 1 if the anchor is positive, otherwise 0. t_i stores four parameterized coordinates of the predicted bounding box while t_i^* is related to the ground-truth box overlapping with a positive anchor. L_{cls} is a binary log loss and L_{reg} is a smoothed L_1 loss similar to (2). These two terms are normalized with the minibatch size (N_{cls}) and the number of anchor locations (N_{reg}), respectively. In the form of FCNs, Faster R-CNN can be trained end-to-end by backpropagation and SGD in an alternate training manner.

With the proposal of Faster R-CNN, region proposal-based CNN architectures for object detection can really be trained in an end-to-end way. Also, a frame rate of 5 frames per second (fps) on a GPU is achieved with the state-of-the-art object detection accuracy on PASCAL VOC 2007 and 2012. However, the alternate training algorithm is very time-consuming and RPN produces objectlike regions (including backgrounds) instead of object instances and is not skilled in dealing with objects with extreme scales or shapes.

5) R-FCN: Divided by the RoI pooling layer, a prevalent family [16], [17] of deep networks for object detection is composed of two subnetworks: a shared fully convolutional subnetwork (independent of RoIs) and an unshared RoI-wise subnetwork. This decomposition originates from pioneering classification architectures (e.g., AlexNet [6] and VGG16 [46]) which consist of a convolutional subnetwork and several FC layers separated by a specific spatial pooling layer.

Recent state-of-the-art image classification networks, such as ResNets [47] and GoogLeNets [45], [93], are fully convolutional. To adapt to these architectures, it is natural to construct a fully convolutional object detection network without RoI-wise subnetwork. However, it turns out to be inferior with such a naive solution [47]. This inconsistency is due to the dilemma of respecting translation variance in object detection compared with increasing translation invariance in image classification. In other words, shifting an object inside an image should be indiscriminative in image classification while any translation of an object in a bounding box may be meaningful in object detection. A manual insertion of the RoI pooling layer into convolutions can break down translation invariance at the expense of additional unshared regionwise layers. Therefore, Dai *et al.* [65] proposed an R-FCNs (see Fig. S2 in the supplementary material).

Different from Faster R-CNN, for each category, the last conv layer of R-FCN produces a total of k^2 position-sensitive score maps with a fixed grid of $k \times k$ first and a position-sensitive RoI pooling layer is then appended to aggregate the responses from these score maps. Finally, in each RoI, k^2 position-sensitive scores are averaged to produce a $C + 1$-d vector and softmax responses across categories are computed. Another $4k^2$-d conv layer is appended to obtain class-agnostic BBs.

With R-FCN, more powerful classification networks can be adopted to accomplish object detection in a fully convolutional architecture by sharing nearly all the layers, and the state-of-the-art results are obtained on both PASCAL VOC and Microsoft COCO [94] data sets at a test speed of 170 ms per image.

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] [Fig. 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 [Fig. 7(b)], and image pyramids are built at test time which results in an inconsistency between train/test-time inferences [16], [17]. 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 [Fig. 7(c)]. To avoid using low-level features, pioneer works [71], [95] usually build the pyramid starting from middle layers or just sum transformed

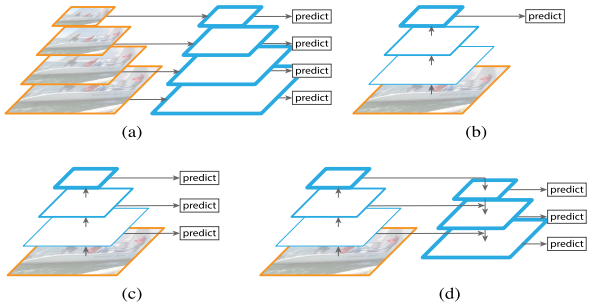


Fig. 7. Main concern of FPN [66]. (a) It is slow to use an image pyramid to build a feature pyramid. (b) Only single-scale features are adopted for faster detection. (c) Alternative to the featurized image pyramid is to reuse the pyramidal feature hierarchy computed by a ConvNet. (d) FPN integrates both (b) and (c). Blue outlines indicate feature maps and thicker outlines denote semantically stronger features.

feature responses, missing the higher resolution maps of the feature hierarchy.

Different from these approaches, FPN [66] holds an architecture with a BU pathway, a top-down (TD) pathway and several lateral connections to combine low-resolution and semantically strong features with high-resolution and semantically weak features [Fig. 7(d)]. The BU pathway, which is the basic forward backbone ConvNet, produces a feature hierarchy by downsampling the corresponding feature maps with a stride of 2. The layers owning the same size of output maps are grouped into the same network stage and the output of the last layer of each stage is chosen as the reference set of feature maps to build the following TD pathway.

To build the TD pathway, feature maps from higher network stages are upsampled at first and then enhanced with those of the same spatial size from the BU pathway via lateral connections. A 1×1 conv layer is appended to the upsampled map to reduce channel dimensions and the mergence is achieved by elementwise addition. Finally, a 3×3 convolution is also appended to each merged map to reduce the aliasing effect of upsampling and the final feature map is generated. This process is iterated until the finest resolution map is generated.

As feature pyramid can extract rich semantics from all levels and be trained end to end with all scales, the state-of-the-art representation can be obtained without sacrificing speed and memory. Meanwhile, FPN is independent of the backbone CNN architectures and can be applied to different stages of object detection (e.g., region proposal generation) and to many other computer vision tasks (e.g., instance segmentation).

7) Mask R-CNN: Instance segmentation [96] is a challenging task which requires detecting all objects in an image and segmenting each instance (semantic segmentation [97]). These two tasks are usually regarded as two independent processes. The multitask scheme will create spurious edge and exhibit systematic errors on overlapping instances [98]. To solve this problem, parallel to the existing branches in Faster R-CNN for classification and bounding box regression, the Mask R-CNN [67] adds a branch to predict segmentation masks in a pixel-to-pixel manner (Fig. 8).

Different from the other two branches that are inevitably collapsed into short output vectors by FC layers, the segmentation mask branch encodes an $m \times m$ mask to maintain the explicit object spatial layout. This kind of fully convolutional

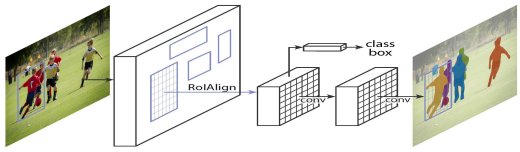


Fig. 8. Mask R-CNN framework for instance segmentation [67].

representation requires fewer parameters but is more accurate than that in [97]. Formally, besides the two losses in (1) for classification and bounding box regression, an additional loss for segmentation mask branch is defined to reach a multitask loss. This loss is only associated with ground-truth class and relies on the classification branch to predict the category.

Because RoI pooling, the core operation in Faster R-CNN, performs a coarse spatial quantization for feature extraction, misalignment is introduced between the RoI and the features. It affects classification little because of its robustness to small translations. However, it has a large negative effect on pixel-to-pixel mask prediction. To solve this problem, Mask R-CNN adopts a simple and quantization-free layer, namely, RoIAlign, to preserve the explicit per-pixel spatial correspondence faithfully. RoIAlign is achieved by replacing the harsh quantization of RoI pooling with bilinear interpolation [99], computing the exact values of the input features at four regularly sampled locations in each RoI bin. In spite of its simplicity, this seemingly minor change improves mask accuracy greatly, especially under strict localization metrics.

Given the Faster R-CNN framework, the mask branch only adds a small computational burden and its cooperation with other tasks provides complementary information for object detection. As a result, Mask R-CNN is simple to implement with promising instance segmentation and object detection results. In a word, Mask R-CNN is a flexible and efficient framework for instance-level recognition, which can be easily generalized to other tasks (e.g., human pose estimation [7], [S4]) with minimal modification.

8) Multitask Learning, Multiscale Representation, and Contextual Modeling: Although the Faster R-CNN gets promising results with several hundred proposals, it still struggles in small-size object detection and localization, mainly due to the coarseness of its feature maps and limited information provided in particular candidate boxes. The phenomenon is more obvious on the Microsoft COCO data set which consists of objects at a broad range of scales, less prototypical images, and requires more precise localization. To tackle these problems, it is of necessity to accomplish object detection with multitask learning [100], multiscale representation [95], and context modeling [101] to combine complementary information from multiple sources.

Multitask learning learns a useful representation for multiple correlated tasks from the same input [102], [103]. Brahmbhatt *et al.* [100] introduced conv features trained for object segmentation and "stuff" (amorphous categories such as ground and water) to guide accurate object detection of small objects (StuffNet). Dai *et al.* [97] presented multitask network cascades of three networks, namely, class-agnostic region proposal generation, pixel-level instance segmentation, and regional instance classification. Li *et al.* [104] incorporated the

weakly supervised object segmentation cues and region-based object detection into a multistage architecture to fully exploit the learned segmentation features.

Multiscale representation combines activations from multiple layers with skip-layer connections to provide semantic information of different spatial resolutions [66]. Cai *et al.* [105] proposed the multiscale CNN (MS-CNN) to ease the inconsistency between the sizes of objects and receptive fields with multiple scale-independent output layers. Yang *et al.* [34] investigated two strategies, namely, scale-dependent pooling (SDP) and layerwise cascaded rejection classifiers (CRCs), to exploit appropriate scale-dependent conv features. Kong *et al.* [101] proposed the HyperNet to calculate the shared features between RPN and object detection network by aggregating and compressing hierarchical feature maps from different resolutions into a uniform space.

Contextual modeling improves detection performance by exploiting features from or around RoIs of different support regions and resolutions to deal with occlusions and local similarities [95]. Zhu *et al.* [106] proposed the SegDeepM to exploit object segmentation which reduces the dependency on initial candidate boxes with the Markov random field. Moysset *et al.* [108] took advantage of four directional 2-D long short-term memories (LSTMs) [107] to convey global context between different local regions and reduced trainable parameters with local parameter sharing. Zeng *et al.* [109] proposed a novel gated bidirectional-net (GBD-Net) by introducing gated functions to control message transmission between different support regions.

The combination incorporates different components above into the same model to improve detection performance further. Gidaris and Komodakis [110] proposed the multiregion CNN (MR-CNN) model to capture different aspects of an object, the distinct appearances of various object parts, and semantic segmentation-aware features. To obtain contextual and multiscale representations, Bell *et al.* [95] proposed the inside–outside net (ION) by exploiting information both inside and outside the RoI with spatial recurrent neural networks [111] and skip pooling [101]. Zagoruyko *et al.* [112] proposed the MultiPath architecture by introducing three modifications to the Fast R-CNN, including multiscale skip connections [95], a modified foveal structure [110], and a novel loss function summing different intersection over union (IoU) losses.

9) Thinking in Deep Learning-Based Object Detection: Apart from the above-mentioned approaches, there are still many important factors for continued progress.

There is a large imbalance between the number of annotated objects and background examples. To address this problem, Shrivastava *et al.* [113] proposed an effective online mining algorithm (OHEM) for automatic selection of the hard examples, which leads to a more effective and efficient training.

Instead of concentrating on feature extraction, Ren *et al.* [114] made a detailed analysis on object classifiers and found that it is of particular importance for object detection to construct a deep and convolutional per-region classifier carefully, especially for ResNets [47] and GoogLeNets [45].

Traditional CNN framework for object detection is not skilled in handling significant scale variation, occlusion, or truncation, especially when only 2-D object detection is involved. To address this problem, Xiang *et al.* [60] proposed a novel subcategory-aware RPN, which guides the generation of region proposals with subcategory information related to object poses and jointly optimize object detection and subcategory classification.

Ouyang *et al.* [115] found that the samples from different classes follow a long-tailed distribution, which indicates that different classes with distinct numbers of samples have different degrees of impacts on feature learning. To this end, objects are first clustered into visually similar class groups, and then, a hierarchical feature learning scheme is adopted to learn deep representations for each group separately.

In order to minimize the computational cost and achieve the state-of-the-art performance, with the "deep and thin" design principle and following the pipeline of Fast R-CNN, Hong *et al.* [116] proposed the architecture of PVANET, which adopts some building blocks including concatenated ReLU [117], Inception [45], and HyperNet [101] to reduce the expense on multiscale feature extraction and trains the network with BN [43], residual connections [47], and learning rate scheduling based on plateau detection [47]. The PVANET achieves the state-of-the-art performance and can be processed in real time on Titan X GPU (21 fps).

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 the 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 the real-time 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 first review some pioneer CNN models and then focus on two significant frameworks, namely, YOLO [18] and 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.* [118] formulated the object detection task as a DNN-based regression, 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 BBs generated by direct upsampling is far from perfect.

Pinheiro *et al.* [119] 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. Inference is efficient since class scores and segmentation can be obtained in a single model with most of the CNN operations shared.

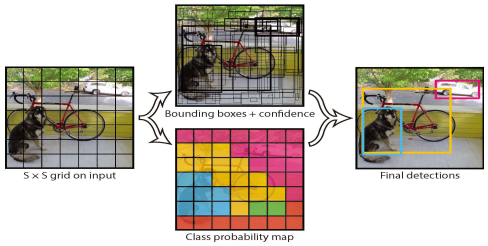


Fig. 9. Main idea of YOLO [18].

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

Yoo *et al.* [69] adopted an iterative classification approach to handle object detection and proposed an impressive end-to-end CNN architecture named AttentionNet. Starting from the top-left and bottom-right corners 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.* [70] proposed a proposal-free iterative grid-based object detector (G-CNN), which models object detection as finding a path from a fixed grid to boxes tightly surrounding the objects [70]. Starting with a fixed multiscale bounding box grid, G-CNN trains a regressor to move and scale elements of the grid toward objects iteratively. However, G-CNN has a difficulty in dealing with small or highly overlapping objects.

2) YOLO: Redmon *et al.* [18] proposed a novel framework called YOLO, which makes the use of the whole topmost feature map to predict both confidences for multiple categories and BBs. The basic idea of YOLO is exhibited in Fig. 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 BBs and their corresponding confidence scores. Formally, confidence scores are defined as $\Pr(\text{Object})*\text{IOU}_{\text{pred}}^{\text{truth}}$, which indicates how likely there exist objects ($\Pr(\text{Object}) \geq 0$) and shows confidences of its prediction ($\text{IOU}_{\text{pred}}^{\text{truth}}$). At the same time, regardless of the number of boxes, C conditional class probabilities ($\Pr(\text{Class}_i|\text{Object})$) 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:

$$\Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} * \Pr(\text{Class}_{\text{i}}|\text{Object}) = \Pr(\text{Class}_{\text{i}}) * \text{IOU}_{\text{pred}}^{\text{truth}} \quad (5)$$

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:

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

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 the confidence scores, $\mathbb{1}_i^{\text{obj}}$ indicates the existence of objects, and $\mathbb{1}_{ij}^{\text{obj}}$ 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 the 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 multiscale 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 [18]. 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.* [71] proposed an SSD, which was inspired by the anchors adopted in MultiBox [68], RPN [17], and multiscale representation [95]. Given a specific feature map, instead of fixed grids adopted in YOLO, the SSD takes the advantage of a set of default anchor boxes with different aspect ratios and scales to discretize the output space of BBs. To handle objects with various sizes, the network fuses predictions from multiple feature maps with different resolutions.

The architecture of SSD is demonstrated in Fig. 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

TABLE I
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 [17]	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 [18]	-	-	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.

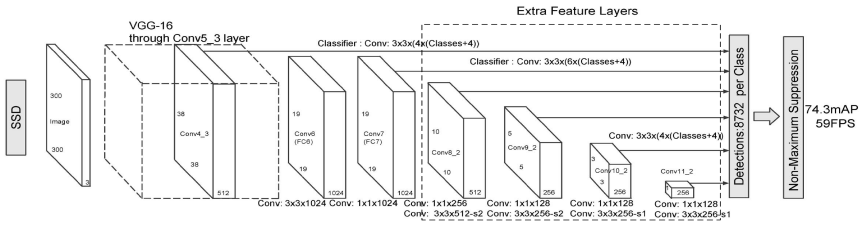


Fig. 10. Architecture of SSD 300 [71]. SSD adds several feature layers to the end of VGG16 backbone network to predict the offsets to default anchor boxes and their associated confidences. Final detection results are obtained by conducting NMS on multiscale refined BBs.

(e.g., Smooth L1) and confidence loss (e.g., Softmax), which is similar to (1). Final detection results are obtained by conducting NMS on multiscale refined BBs.

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].

C. Experimental Evaluation

We compare various object detection methods on three benchmark data sets, 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], networks on convolutional feature maps (NOC) [114], Bayes [85], MR-CNN& S-CNN [105], Faster R-CNN [17], HyperNet [101], ION [95], MS-GR [104], StuffNet [100], SSD300 [71], SSD512 [71], OHEM [113], SDP+CRC [34], G-CNN [70], SubCNN [60], GBD-Net [109], PVANET [116], YOLO [18], 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 the paper length, we only provide an overview, including proposal, learning method, loss function, programing 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 data sets consist of 20 categories. The evaluation terms are AP in each single category and mAP across all the 20 categories. Comparative results are exhibited in Tables II and III, from which the following remarks can be obtained.

1) If incorporated with a proper way, more powerful backbone CNN models can definitely improve the object detection performance (the comparison among R-CNN with AlexNet, R-CNN with VGG16 and SPP-net with ZF-Net [122]).
2) With the introduction of the SPP layer (SPP-net), end-to-end multitask architecture (FRCN), and RPN (Faster R-CNN), object detection performance is improved gradually and apparently.
3) Due to a large number of trainable parameters, in order to obtain multilevel robust features, data augmentation is very important for deep learning-based models (Faster R-CNN with "07," "07 + 12," and "07 + 12 + coco").
4) Apart from basic models, there are still many other factors affecting object detection performance, such as multiscale and multiregion feature extraction (e.g., MR-CNN), modified classification networks (e.g., NOC), additional information from other correlated tasks (e.g., StuffNet, HyperNet), multiscale representation (e.g., ION), and mining of hard negative samples (e.g., OHEM).
5) 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).
6) By combining many recent tricks and modeling the whole network as a fully convolutional one, R-FCN

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]	07	73.4	77.0	63.4	45.4	44.6	75.1	78.1	79.8	40.5	73.7	62.2	79.4	78.1	73.1	64.2	35.6	66.8	67.2	70.4	71.1	66.0
SPP-net(ZF) [64]	07	68.5	71.7	58.7	41.9	42.5	67.7	72.1	73.8	34.7	67.0	63.4	66.0	72.5	71.3	58.9	32.8	60.9	56.1	67.9	68.8	60.9
GCNN [70]	07	68.3	77.3	68.5	52.4	38.6	78.5	79.5	81.0	47.1	73.6	64.5	77.2	80.5	75.8	66.6	34.3	65.2	64.4	75.6	66.4	66.8
Bayes [85]	07	74.1	83.2	67.0	50.8	51.6	76.2	81.4	77.2	48.1	78.9	65.6	77.3	78.4	75.1	70.1	41.4	69.6	60.8	70.2	73.7	68.5
Fast R-CNN [16]	07+12	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4	70.0
SDP+CRC [34]	07	76.1	79.4	68.2	52.6	46.0	78.4	78.4	81.0	46.7	73.5	65.3	78.6	81.0	76.7	77.3	39.0	65.1	67.2	77.5	70.3	68.9
SubCNN [60]	07	70.2	80.5	69.5	60.3	47.9	79.0	78.7	84.2	48.5	73.9	63.0	82.7	80.6	76.0	70.2	38.2	62.4	67.7	77.7	60.5	68.5
StuffNet30 [100]	07	72.6	81.7	70.6	60.5	53.0	81.5	83.7	83.9	52.2	78.9	70.7	85.0	85.7	77.0	78.7	42.2	73.6	69.2	79.2	73.8	72.7
NOC [114]	07+12	76.3	81.4	74.4	61.7	60.8	84.7	78.2	82.9	53.0	79.2	69.2	83.2	83.2	78.5	68.0	45.0	71.6	76.7	82.2	75.7	73.3
MR-CNN&S-CNN [105]	07+12	80.3	84.1	78.5	70.8	**68.5**	88.0	85.9	87.8	60.3	85.2	73.7	87.2	86.5	85.0	76.4	48.5	76.3	75.5	85.0	81.0	78.2
HyperNet [101]	07+12	77.4	83.3	75.0	69.1	62.4	83.1	87.4	87.4	57.1	79.8	71.4	85.1	85.1	80.0	79.1	51.2	79.1	75.7	80.9	76.5	76.3
MS-GR [104]	07+12	80.0	81.0	77.4	72.1	64.3	88.2	88.1	88.4	64.4	85.4	73.1	87.3	87.4	85.1	79.6	50.1	78.4	79.5	86.9	75.5	78.6
OHEM+Fast R-CNN [113]	07+12	80.6	85.7	79.8	69.9	60.8	88.3	87.9	89.6	59.7	85.1	76.5	87.1	87.3	82.4	78.8	53.7	80.5	78.7	84.5	80.7	78.9
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 [17]	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 [17]	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 [17]	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+COCO	**86.6**	**88.3**	**82.4**	76.0	66.3	**88.6**	**88.9**	89.1	**65.1**	**88.4**	73.6	86.5	88.9	**85.3**	**84.6**	**59.1**	**85.0**	80.4	**87.4**	81.2	**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 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]	12	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1	53.3
R-CNN(VGG16) [15]	12	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3	62.4
Bayes [85]	12	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2	66.4
Fast R-CNN [65]	07++12	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2	68.4
SutffNet30 [100]	12	83.0	76.9	71.2	51.6	50.1	76.4	75.7	87.8	48.3	74.8	55.7	85.7	81.2	80.3	79.5	44.2	71.8	61.0	78.5	65.4	70.0
NOC [114]	07+12	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1	68.8
MR-CNN&S-CNN [105]	07++12	85.5	82.9	76.6	57.8	62.7	79.4	77.2	86.6	55.0	79.1	62.2	87.0	83.4	84.7	78.9	45.3	73.4	65.8	80.3	74.0	73.9
HyperNet [101]	07++12	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7	71.4
OHEM+Fast R-CNN [113]	07++12+coco	90.1	87.4	79.9	65.8	66.3	86.1	85.0	92.9	62.4	83.4	69.5	90.6	88.9	88.9	83.6	59.0	82.0	74.7	88.2	77.3	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 [17]	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 [17]	07++12+coco	87.4	83.6	76.8	62.9	59.6	81.9	82.0	91.3	54.9	82.6	59.0	89.0	85.5	84.7	84.1	52.2	78.9	65.5	85.4	70.2	75.9
YOLO [18]	07++12	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8	57.9
YOLO+Fast R-CNN [18]	07++12	83.4	78.5	73.5	55.8	43.4	79.1	73.1	89.4	49.4	75.5	57.0	87.5	80.9	81.0	74.7	41.8	71.5	68.5	82.1	67.2	70.7
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+coco	91.4	88.6	82.6	71.4	63.1	**87.4**	88.1	93.9	66.9	86.6	66.3	92.0	91.7	90.8	88.5	60.9	87.0	75.4	90.2	80.4	82.2
R-FCN (ResNet101) [65]	07++12+coco	**92.3**	**89.9**	**86.7**	**74.7**	**75.2**	86.7	**89.0**	**95.8**	**70.2**	**90.4**	66.5	**95.0**	**93.2**	**92.1**	**91.1**	**71.0**	**89.7**	**76.0**	**92.0**	**83.4**	**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.

achieves a 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 data set 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 given in Table IV.

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

1) Multiscale training and test are beneficial in improving object detection performance, which provide additional information in different resolutions (R-FCN). FPN and DSSD provide some better ways to build feature pyramids to achieve multiscale representation. The complementary information from other related tasks is also helpful for accurate object localization (Mask R-CNN with instance segmentation task).
2) Overall, region proposal-based methods, such as Faster R-CNN and R-FCN, perform better than regression/classification-based approaches, namely, YOLO and SSD, due to the fact that quite a lot of localization errors are produced by regression/classification-based approaches.
3) Context modeling is helpful to locate small objects, which provides additional information by consulting nearby objects and surroundings (GBD-Net and multipath).
4) Due to the existence of a large number of nonstandard small objects, the results on this data set are much worse than those of VOC 2007/2012. With the introduction of other powerful frameworks (e.g., ResNeXt [123]) and useful strategies (e.g., multitask learning [67], [124]), the performance can be improved.
5) The success of DSOD in training from scratch stresses the importance of the network design to release the requirements for perfect pretrained classifiers on relevant tasks and a large number of annotated samples.

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

1) By computing CNN features on shared feature maps (SPP-net), test consumption is reduced largely. Test time is further reduced with the unified multitask learning (FRCN) and removal of additional region proposal generation stage (Faster R-CNN). It is also helpful to compress the parameters of FC layers with SVD [91] (PAVNET and FRCN).
2) It takes additional test time to extract multiscale features and contextual information (ION and MR-RCNN& S-RCNN).
3) It takes more time to train a more complex and deeper network (ResNet101 against VGG16) and this time consumption can be reduced by adding as many layers into shared fully convolutional layers as possible (FRCN).

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 [17]	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.

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 [34]	07	68.9	0.47	2.1
SS+HyperNet* [101]	07+12	76.3	0.20	5
MR-CNN&S-CNN [105]	07+12	78.2	30	0.03
ION [95]	07+12+S	79.2	1.92	0.5
Faster R-CNN(VGG16) [17]	07+12	73.2	0.11	9.1
Faster R-CNN(ResNet101) [17]	07+12	**83.8**	2.24	0.4
YOLO [18]	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.

4) Regression-based models can usually be processed in real time 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] and residual connections [123].

IV. SALIENT OBJECT DETECTION

Visual saliency detection, one of the most important and challenging tasks in computer vision, aims to highlight the most dominant object regions in an image. Numerous applications incorporate the visual saliency to improve their performance, such as image cropping [125] and segmentation [126], image retrieval [57], and object detection [66].

Broadly, there are two branches of approaches in salient object detection, namely, BU [127] and TD [128]. Local feature contrast plays the central role in BU salient object detection, regardless of the semantic contents of the scene. To learn local feature contrast, various local and global features are extracted from pixels, e.g., edges [129] and spatial information [130]. However, high-level and multiscale semantic information cannot be explored with these low-level features. As a result, low-contrast salient maps instead of salient objects are obtained. TD salient object detection is task-oriented and takes prior knowledge about object categories to guide the generation of salient maps. Taking semantic segmentation as an example, a saliency map is generated in the segmentation to assign pixels to particular object categories via a TD approach [131]. In a word, TD saliency can be viewed as a focus-of-attention mechanism, which prunes BU salient points that are unlikely to be parts of the object [132].

A. Deep Learning in Salient Object Detection

Due to the significance for providing high-level and multiscale feature representation and the successful applications in many correlated computer vision tasks, such as semantic segmentation [131], edge detection [133], and generic object detection [16], it is feasible and necessary to extend CNN to salient object detection.

The early work by Vig *et al.* [29] follows a completely automatic data-driven approach to perform a large-scale search for optimal features, namely, an ensemble of deep networks with different layers and parameters. To address the problem of limited training data, Kummerer *et al.* [134] proposed the Deep Gaze by transferring from the AlexNet to generate a high-dimensional feature space and create a saliency map. A similar architecture was proposed by Huang *et al.* [135] to integrate saliency prediction into pretrained object recognition DNNs. The transfer is accomplished by fine-tuning DNNs' weights with an objective function based on the saliency evaluation metrics, such as similarity, KL-divergence, and normalized scanpath saliency.

Some works combined local and global visual clues to improve salient object detection performance. Wang *et al.* [136] trained two independent deep CNNs (DNN-L and DNN-G) to capture local information and global contrast and predicted saliency maps by integrating both local estimation and global search. Cholakkal *et al.* [137] proposed a weakly supervised saliency detection framework to combine visual saliency from BU and TD saliency

TABLE VI
COMPARISON BETWEEN STATE-OF-THE-ART METHODS

Dataset	Metrics	CHM [150]	RC [151]	DRFI [152]	MC [138]	MDF [146]	LEGS [136]	DSR [149]	MTDNN [141]	CRPSD [142]	DCL [143]	ELD [153]	NLDF [154]	DSSC [155]
PASCAL-S	wF_β	0.631	0.640	0.679	0.721	0.764	0.756	0.697	0.818	0.776	0.822	0.767	**0.831**	0.830
	MAE	0.222	0.225	0.221	0.147	0.145	0.157	0.128	0.170	**0.063**	0.108	0.121	0.099	0.080
ECSSD	wF_β	0.722	0.741	0.787	0.822	0.833	0.827	0.872	0.810	0.849	0.898	0.865	0.905	**0.915**
	MAE	0.195	0.187	0.166	0.107	0.108	0.118	**0.037**	0.160	0.046	0.071	0.098	0.063	0.052
HKU-IS	wF_β	0.728	0.726	0.783	0.781	0.860	0.770	0.833	-	0.821	0.907	0.844	0.902	**0.913**
	MAE	0.158	0.165	0.143	0.098	0.129	0.118	0.040	-	0.043	0.048	0.071	0.048	**0.039**
SOD	wF_β	0.655	0.657	0.712	0.708	0.785	0.707	-	0.781	-	0.832	0.760	0.810	**0.842**
	MAE	0.249	0.242	0.215	0.184	0.155	0.205	-	0.150	-	0.126	0.154	0.143	**0.118**

* The bigger wF_β is or the smaller MAE is, the better the performance is.

maps and refined the results with a multiscale superpixel-averaging. Zhao *et al.* [138] proposed a multicontext deep learning framework, which utilizes a unified learning framework to model global and local context jointly with the aid of superpixel segmentation. To predict saliency in videos, Bak *et al.* [139] fused two static saliency models, namely, spatial stream net and temporal stream net, into a two-stream framework with a novel empirically grounded data augmentation technique.

Complementary information from semantic segmentation and context modeling is beneficial. To learn internal representations of saliency efficiently, He *et al.* [140] proposed a novel superpixelwise CNN approach called SuperCNN, in which salient object detection is formulated as a binary labeling problem. Based on a fully CNN, Li *et al.* [141] proposed a multitask deep saliency model, in which intrinsic correlations between saliency detection and semantic segmentation are set up. However, due to the conv layers with large receptive fields and pooling layers, blurry object boundaries and coarse saliency maps are produced. Tang and Wu [142] proposed a novel saliency detection framework (CRPSD) [142], which combines the region-level saliency estimation and pixel-level saliency prediction together with three closely related CNNs. Li and Yu [143]proposed a deep contrast network to combine segmentwise spatial pooling and pixel-level fully convolutional streams [143].

The proper integration of multiscale feature maps is also of significance for improving detection performance. Based on Fast R-CNN, Wang *et al.* [144] proposed the RegionNet by performing salient object detection with end-to-end edge preserving and multiscale contextual modeling. Liu *et al.* [28] proposed a multiresolution CNN (Mr-CNN) to predict eye fixations, which is achieved by learning both BU visual saliency and TD visual factors from raw image data simultaneously. Cornia *et al.* [145] proposed an architecture that combines features extracted at different levels of the CNN. Li and Yu [146] proposed a multiscale deep CNN framework to extract three scales of deep contrast features, namely, the mean-subtracted region, the bounding box of its immediate neighboring regions, and the masked entire image, from each candidate region.

It is efficient and accurate to train a direct pixelwise CNN architecture to predict salient objects with the aids of recurrent neural networks and deconvolution networks. Pan *et al.* [147] formulated saliency prediction as a minimization optimization on the Euclidean distance between the predicted saliency map and the ground truth and proposed two kinds of architectures: a shallow one trained from scratch and a deeper one adapted from a deconvoluted VGG network. Asconvolutional–deconvolution networks are not expert in recognizing objects of multiple scales, Kuen *et al.* [148] proposed a recurrent attentional convolutional–deconvolution network with several spatial transformer and recurrent network units to conquer this problem. To fuse local, global, and contextual information of salient objects, Tang *et al.* [149] developed a deeply supervised recurrent CNN to perform a full image-to-image saliency detection.

B. Experimental Evaluation

Four representative data sets, including Evaluation on Complex Scene Saliency Dataset (ECSSD) [156], HKU-IS [146], PASCALS [157], and SOD [158], are used to evaluate several state-of-the-art methods. ECSSD consists of 1000 structurally complex but semantically meaningful natural images. HKU-IS is a large-scale data set containing over 4000 challenging images. Most of these images have more than one salient object and own low contrast. PASCALS is a subset chosen from the validation set of PASCAL VOC 2010 segmentation data set and is composed of 850 natural images. The SOD data set possesses 300 images containing multiple salient objects. The training and validation sets for different data sets are kept the same as those in [152].

Two standard metrics, namely, F-measure and the mean absolute error (MAE), are utilized to evaluate the quality of a saliency map. Given precision and recall values precomputed on the union of generated binary mask B and ground truth Z, F-measure is defined as follows:

$$F_\beta = \frac{(1+\beta^2)\text{Presion} \times \text{Recall}}{\beta^2 \text{Presion} + \text{Recall}} \tag{7}$$

where β^2 is set to 0.3 in order to stress the importance of the precision value.

The MAE score is computed with the following equation:

$$\text{MAE} = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} |\hat{S}(i,j) = \hat{Z}(i,j)| \tag{8}$$

where $\hat{Z}$ and $\hat{S}$ represent the ground truth and the continuous saliency map, respectively. W and H are the width and height of the salient area, respectively. This score stresses the importance of successfully detected salient objects over detected nonsalient pixels [159].

The following approaches are evaluated: contextual hypergraph modeling (CHM) [150], RC [151], discriminative

regional feature integration (DRFI) [152], MC [138], multiscale deep CNN features (MDF) [146], local estimation and global search (LEGS) [136], DSR [149], multi-task deep neural network [141], CRPSD [142], deep contrast learning (DCL) [143], encoded low level distance (ELD) [153], nonlocal deep features (NLDF) [154], and deep supervision with short connections (DSSC) [155]. Among these methods, CHM, RC, and DRFI are classical ones with the best performance [159], while the other methods are all associated with CNN. F-measure and MAE scores are given in Table VI.

From Table VI, we can find that CNN-based methods perform better than classic methods. MC and MDF combine the information from local and global context to reach a more accurate saliency. ELD refers to low-level handcrafted features for complementary information. LEGS adopts generic region proposals to provide initial salient regions, which may be insufficient for salient detection. DSR and MT act in different ways by introducing a recurrent network and semantic segmentation, which provide insights for future improvements. CRPSD, DCL, NLDF, and DSSC are all based on multiscale representations and superpixel segmentation, which provide robust salient regions and smooth boundaries. DCL, NLDF, and DSSC perform the best on these four data sets. DSSC earns the best performance by modeling scale-to-scale short connections.

Overall, as CNN mainly provides salient information in local regions, most of the CNN-based methods need to model visual saliency along region boundaries with the aid of superpixel segmentation. Meanwhile, the extraction of multiscale deep CNN features is of significance for measuring local conspicuity. Finally, it is necessary to strengthen local connections between different CNN layers as well as to utilize complementary information from local and global context.

V. Face Detection

Face detection is essential to many face applications and acts as an important preprocessing procedure to face recognition [160]–[162], face synthesis [163], [164], and facial expression analysis [165]. Different from generic object detection, this task is to recognize and locate face regions covering a very large range of scales (30–300 pts versus 10–1000 pts). At the same time, faces have their unique object structural configurations (e.g., the distribution of different face parts) and characteristics (e.g., skin color). All these differences lead to special attention to this task. However, large visual variations of faces, such as occlusions, pose variations, and illumination changes, impose great challenges for this task in real applications.

The most famous face detector proposed by Viola and Jones [166] trains cascaded classifiers with Haar-like features and AdaBoost, achieving good performance with real-time efficiency. However, this detector may degrade significantly in real-world applications due to larger visual variations of human faces. Different from this cascade structure, Felzenszwalb *et al.* [24] proposed a deformable part model (DPM) for face detection. However, for these traditional face detection methods, high computational expenses and large quantities of annotations are required to achieve a reasonable result.

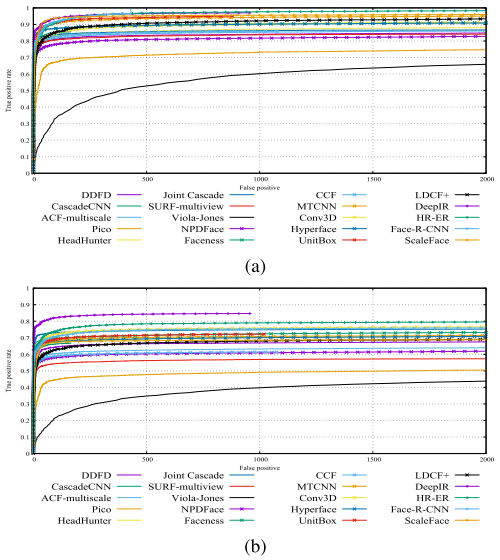


Fig. 11. ROC curves of state-of-the-art methods on FDDB. (a) Discrete ROC curves. (b) Continuous ROC curves.

In addition, their performance is greatly restricted by manually designed features and shallow architecture.

A. Deep Learning in Face Detection

Recently, some CNN-based face detection approaches have been proposed [167]–[169]. As less accurate localization results from independent regressions of object coordinates, Yu *et al.* [167] proposed a novel IoU loss function for predicting the four bounds of box jointly. Farfade *et al.* [168] proposed a deep dense face detector (DDFD) to conduct multiview face detection, which is able to detect faces in a wide range of orientations without the requirement of pose/landmark annotations. Yang *et al.* [169] proposed a novel deep learning-based face detection framework, which collects the responses from local facial parts (e.g., eyes, nose, and mouths) to address face detection under severe occlusions and unconstrained pose variations. Yang *et al.* [170] proposed a scale-friendly detection network named ScaleFace, which splits a large range of target scales into smaller subranges. Different specialized subnetworks are constructed on these subscales and combined into a single one to conduct end-to-end optimization. Hao *et al.* [171] designed an efficient CNN to predict the scale distribution histogram of the faces and took this histogram to guide the zoomed-in view and zoomed-out view of the image. Since the faces are approximately in uniform scale after zoom, compared with other state-of-the-art baselines, better performance is achieved with a less computation cost. In addition, some generic detection frameworks are extended to face detection with different modifications, e.g., Faster R-CNN [30], [172], [173].

Some authors trained CNNs with other complementary tasks, such as 3-D modeling and face landmarks, in a multitask learning manner. Huang *et al.* [174] proposed a unified end-to-end FCN framework called DenseBox to jointly conduct face detection and landmark localization. Li *et al.* [175] proposed a multitask discriminative learning framework that integrates a ConvNet with a fixed 3-D mean face model in an end-to-end

manner. In the framework, two issues are addressed to transfer from generic object detection to face detection, namely, eliminating predefined anchor boxes by a 3-D mean face model and replacing RoI pooling layer with a configuration pooling layer. Zhang *et al.* [176] proposed a deep cascaded multitask framework named multitask cascaded convolutional networks (MTCNN) which exploits the inherent correlations between face detection and alignment in the unconstrained environment to boost up detection performance in a coarse-to-fine manner.

Reducing computational expenses is of necessity in real applications. To achieve real-time detection on the mobile platform, Kalinovskii and Spitsyn [177] proposed a new solution of frontal face detection based on compact CNN cascades. This method takes a cascade of three simple CNNs to generate, classify, and refine candidate object positions progressively. To reduce the effects of large pose variations, Chen *et al.* [32] proposed a cascaded CNN denoted by supervised transformer network. This network takes a multitask RPN to predict candidate face regions along with associated facial landmarks simultaneously and adopts a generic R-CNN to verify the existence of valid faces. Yang and Nevatia [8] proposed a three-stage cascade structure based on FCNs, while in each stage, a multiscale FCN is utilized to refine the positions of possible faces. Qin *et al.* [178] proposed a unified framework that achieves better results with the complementary information from different jointly trained CNNs.

B. Experimental Evaluation

The FDDB [179] data set has a total of 2845 pictures in which 5171 faces are annotated with an elliptical shape. Two types of evaluations are used: the discrete score and continuous score. By varying the threshold of the decision rule, the receiver operating characteristic (ROC) curve for the discrete scores can reflect the dependence of the detected face fractions on the number of false alarms. Compared with annotations, any detection with an IoU ratio exceeding 0.5 is treated as positive. Each annotation is only associated with one detection. The ROC curve for the continuous scores is the reflection of face localization quality.

The evaluated models cover DDFD [168], Cascade-CNN [180], aggregate channel features (ACF)-multiscale [181], Pico [182], Head-Hunter [183], Joint Cascade [31], SURF-multiview [184], Viola–Jones [166], NPDFace [185], Faceness [169], convolutional channel features (CCF) [186], MTCNN [176], Conv3-D [175], Hyperface [187], UnitBox [167], locally decorrelated channel features (LDCF+) [S2], DeepIR [173], hybrid-resolution model with elliptical regressor (HR-ER) [188], Face-R-CNN [172], and ScaleFace [170]. ACF-multiscale, Pico, HeadHunter, Joint Cascade, SURF-multiview, Viola-Jones, NPDFace, and LDCF+ are built on classic hand-crafted features while the rest methods are based on deep CNN features. The ROC curves are shown in Fig. 11.

In Fig. 11(a), in spite of relatively competitive results produced by LDCF+, it can be observed that most of the classic methods perform with similar results and are outperformed by CNN-based methods by a significant margin. In Fig. 11(b), it can be observed that most of the CNN-based methods earn similar true positive rates between 60% and 70% while DeepIR and HR-ER perform much better than them. Among classic methods, Joint Cascade is still competitive. As earlier works, DDFD and CCF directly make use of generated feature maps and obtain relatively poor results. CascadeCNN builds cascaded CNNs to locate face regions, which is efficient but inaccurate. Faceness combines the decisions from different part detectors, resulting in precise face localizations while being time-consuming. The outstanding performance of MTCNN, Conv3-D, and Hyperface proves the effectiveness of multitask learning. HR-ER and ScaleFace adaptively detect faces of different scales and make a balance between accuracy and efficiency. DeepIR and Face-R-CNN are two extensions of the Faster R-CNN architecture to face detection, which validate the significance and effectiveness of Faster R-CNN. Unitbox provides an alternative choice for performance improvements by carefully designing optimization loss.

From these results, we can draw the conclusion that CNN-based methods are in the leading position. The performance can be improved by the following strategies: designing novel optimization loss, modifying generic detection pipelines, building meaningful network cascades, adapting scale-aware detection, and learning multitask shared CNN features.

VI. Pedestrian Detection

Recently, pedestrian detection has been intensively studied, which has a close relationship to pedestrian tracking [189], [190], person reidentification [191], [192], and robot navigation [193], [194]. Prior to the recent progress in deep CNN (DCNN)-based methods [195], [196], some researchers combined boosted decision forests with hand-crafted features to obtain pedestrian detectors [197]–[199]. At the same time, to explicitly model the deformation and occlusion, part-based models [200] and explicit occlusion handling [201], [202] are of concern.

As there are many pedestrian instances of small sizes in typical scenarios of pedestrian detection (e.g., automatic driving and intelligent surveillance), the application of RoI pooling layer in generic object detection pipeline may result in "plain" features due to collapsing bins. In the meantime, the main source of false predictions in pedestrian detection is the confusion of hard background instances, which is in contrast to the interference from multiple categories in generic object detection. As a result, different configurations and components are required to accomplish accurate pedestrian detection.

A. Deep Learning in Pedestrian Detection

Although DCNNs have obtained excellent performance on generic object detection [16], [72], none of these approaches have achieved better results than the best hand-crafted feature-based method [198] for a long time, even when part-based information and occlusion handling are incorporated [202]. Thereby, some studies have been conducted to analyze the reasons. Zhang *et al.* [203] attempted to adapt generic Faster R-CNN [17] to pedestrian detection. They modified the downstream classifier by adding boosted forests to shared, high-resolution conv feature maps and taking an RPN to handle small instances and hard negative examples. To deal with

complex occlusions existing in pedestrian images, inspired by DPM [24], Tian *et al.* [204] proposed a deep learning framework called DeepParts, which makes decisions based on an ensemble of extensive part detectors. DeepParts has advantages in dealing with weakly labeled data, low IoU positive proposals, and partial occlusion.

Other researchers also tried to combine complementary information from multiple data sources. CompACT-Deep adopts a complexity-aware cascade to combine hand-crafted features and fine-tuned DCNNs [195]. Based on Faster R-CNN, Liu *et al.* [205] proposed multispectral DNNs for pedestrian detection to combine complementary information from color and thermal images. Tian *et al.* [206] proposed a task-assistant CNN to jointly learn multiple tasks with multiple data sources and to combine pedestrian attributes with semantic scene attributes together. Du *et al.* [207] proposed a DNN fusion architecture for fast and robust pedestrian detection. Based on the candidate BBs generated with SSD detectors [71], multiple binary classifiers are processed parallelly to conduct soft-rejection-based network fusion by consulting their aggregated degree of confidences.

However, most of these approaches are much more sophisticated than the standard R-CNN framework. CompACT-Deep consists of a variety of hand-crafted features, a small CNN model, and a large VGG16 model [195]. DeepParts contains 45 fine-tuned DCNN models, and a set of strategies, including bounding box shifting handling and part selection, are required to arrive at the reported results [204]. Therefore, the modification and simplification are of significance to reduce the burden on both software and hardware to satisfy real-time detection demand. Tome *et al.* [59] proposed a novel solution to adapt generic object detection pipeline to pedestrian detection by optimizing most of its stages. Hu *et al.* [208] trained an ensemble of boosted decision models by reusing the conv feature maps, and a further improvement was gained with simple pixel labeling and additional complementary hand-crafted features. Tome *et al.* [209] proposed a reduced memory region-based deep CNN architecture, which fuses regional responses from both ACF detectors and SVM classifiers into R-CNN. Ribeiro *et al.* [33] addressed the problem of human-aware navigation and proposed a vision-based person tracking system guided by multiple camera sensors.

B. Experimental Evaluation

The evaluation is conducted on the most popular Caltech Pedestrian data set [3]. The data set was collected from the videos of a vehicle driving through an urban environment and consists of 250 000 frames with about 2300 unique pedestrians and 350 000 annotated BBs. Three kinds of labels, namely, "Person (clear identifications)," "Person? (unclear identifications)," and "People (large group of individuals)," are assigned to different BBs. The performance is measured with the log-average miss rate (L-AMR) which is computed evenly spaced in log-space in the range 10^{-2} to 1 by averaging miss rate at the rate of nine false positives per image [3]. According to the differences in the height and visible part of the BBs, a total of nine popular settings are adopted to evaluate different properties of these models. Details of these settings are as in [3].

Evaluated methods include Checkerboards+ [198], LDCF++ [S2], SCF+AlexNet [210], SA-FastRCNN [211], MS-CNN [105], DeepParts [204], CompACT-Deep [195], RPN+BF [203], and F-DNN+SS [207]. The first two methods are based on hand-crafted features while the rest ones rely on deep CNN features. All results are exhibited in Table VII. From this table, we observe that different from other tasks, classic handcrafted features can still earn competitive results with boosted decision forests [203], ACF [197], and HOG+LUV channels [S2]. As an early attempt to adapt CNN to pedestrian detection, the features generated by SCF+AlexNet are not so discriminant and produce relatively poor results. Based on multiple CNNs, DeepParts and CompACT-Deep accomplish detection tasks via different strategies, namely, local part integration and cascade network. The responses from different local part detectors make DeepParts robust to partial occlusions. However, due to complexity, it is too time-consuming to achieve real-time detection. The multiscale representation of MS-CNN improves the accuracy of pedestrian locations. SA-FastRCNN extends Fast R-CNN to automatically detect pedestrians according to their different scales, which has trouble when there are partial occlusions. RPN+BF combines the detectors produced by Faster R-CNN with boosting decision forest to accurately locate different pedestrians. F-DNN+SS, which is composed of multiple parallel classifiers with soft rejections, performs the best followed by RPN+BF, SA-FastRCNN, and MS-CNN.

In short, CNN-based methods can provide more accurate candidate boxes and multilevel semantic information for identifying and locating pedestrians. Meanwhile, handcrafted features are complementary and can be combined with CNN to achieve better results. The improvements over existing CNN methods can be obtained by carefully designing the framework and classifiers, extracting multiscale and part-based semantic information and searching for complementary information from other related tasks, such as segmentation.

VII. Promising Future Directions and Tasks

In spite of rapid development and achieved promising progress of object detection, there are still many open issues for the future work.

The first one is small object detection such as occurring in COCO data set and in face detection task. To improve localization accuracy on small objects under partial occlusions, it is necessary to modify network architectures from the following aspects.

1) *Multitask Joint Optimization and Multimodal Information Fusion:* Due to the correlations between different tasks within and outside object detection, multitask joint optimization has already been studied by many researchers [16], [17]. However, apart from the tasks mentioned in Section III-A8, it is desirable to think over the characteristics of different subtasks of object detection (e.g., superpixel semantic segmentation in salient object detection) and extend multitask optimization to other applications such as instance segmentation [66], multiobject tracking [202], and multiperson pose estimation [S4]. In addition, given

TABLE VII
DETAILED BREAKDOWN PERFORMANCE COMPARISONS OF STATE-OF-THE-ART MODELS ON CALTECH PEDESTRIAN DATA SET. ALL NUMBERS ARE REPORTED IN L-AMR

Method	Reasonable	All	Far	Medium	Near	none	partial	heavy
Checkerboards+ [198]	17.1	68.4	100	58.3	5.1	15.6	31.4	78.4
LDCF++ [S2]	15.2	67.1	100	58.4	5.4	13.3	33.3	76.2
SCF+AlexNet [210]	23.3	70.3	100	62.3	10.2	20.0	48.5	74.7
SA-FastRCNN [211]	9.7	62.6	100	51.8	**0**	7.7	24.8	64.3
MS-CNN [105]	10.0	61.0	97.2	49.1	2.6	8.2	19.2	60.0
DeepParts [204]	11.9	64.8	100	56.4	4.8	10.6	19.9	60.4
CompACT-Deep [195]	11.8	64.4	100	53.2	4.0	9.6	25.1	65.8
RPN+BF [203]	9.6	64.7	100	53.9	2.3	7.7	24.2	74.2
F-DNN+SS [207]	**8.2**	**50.3**	**77.5**	**33.2**	2.8	**6.7**	**15.1**	**53.4**

a specific application, the information from different modalities, such as text [212], thermal data [205], and images [65], can be fused together to achieve a more discriminant network.

2) *Scale Adaption:* Objects usually exist in different scales, which are more apparent in face detection and pedestrian detection. To increase the robustness to scale changes, it is demanded to train scale-invariant, multiscale or scale-adaptive detectors. For scale-invariant detectors, more powerful backbone architectures (e.g., ResNext [123]), negative sample mining [113], reverse connection [213], and subcategory modeling [60] are all beneficial. For multiscale detectors, both the FPN [66] that produces multiscale feature maps and the generative adversarial network [214] that narrows representation differences between small objects and the large ones with a low-cost architecture provide insights into generating meaningful feature pyramid. For scale-adaptive detectors, it is useful to combine knowledge graph [215], attentional mechanism [216], cascade network [180], and scale distribution estimation [171] to detect objects adaptively.
3) *Spatial Correlations and Contextual Modeling:* Spatial distribution plays an important role in object detection. Therefore, region proposal generation and grid regression are taken to obtain probable object locations. However, the correlations between multiple proposals and object categories are ignored. In addition, the global structure information is abandoned by the position-sensitive score maps in R-FCN. To solve these problems, we can refer to diverse subset selection [217] and sequential reasoning tasks [218] for possible solutions. It is also meaningful to mask salient parts and couple them with the global structure in a joint-learning manner [219].

The second one is to release the burden on manual labor and accomplish real-time object detection, with the emergence of the large-scale image and video data. The following three aspects can be taken into account.

1) *Cascade Network:* In a cascade network, a cascade of detectors is built in different stages or layers [180], [220]. Easily distinguishable examples are rejected at shallow layers so that features and classifiers at later stages can handle more difficult samples with the aid of the decisions from previous stages. However, current cascades are built in a greedy manner, where previous stages in cascade are fixed when training a new stage. Therefore, the optimizations of different CNNs are isolated, which stresses the necessity of end-to-end optimization for CNN cascade. At the same time, it is also a matter of concern to build contextual associated cascade networks with existing layers.
2) *Unsupervised and Weakly Supervised Learning:* It is very time-consuming to manually draw large quantities of BBs. To release this burden, semantic prior [55], unsupervised object discovery [221], multiple instance learning [222], and DNN prediction [47] can be integrated to make the best use of image-level supervision to assign object category tags to corresponding object regions and refine object boundaries. Furthermore, weakly annotations (e.g., center-click annotations [223]) are also helpful for achieving high-quality detectors with modest annotation efforts, especially aided by the mobile platform.
3) *Network Optimization:* Given specific applications and platforms, it is significant to make a balance among speed, memory, and accuracy by selecting an optimal detection architecture [116], [224]. However, despite that detection accuracy is reduced, it is more meaningful to learn compact models with a fewer number of parameters [209]. This situation can be relieved by introducing better pretraining schemes [225], knowledge distillation [226], and hint learning [227]. DSOD also provides a promising guideline to train from scratch to bridge the gap between different image sources and tasks [74].

The third one is to extend typical methods for 2-D object detection to adapt 3-D object detection and video object detection, with the requirements from autonomous driving, intelligent transportation, and intelligent surveillance.

1) *3-D Object Detection:* With the applications of 3-D sensors (e.g., Light Detection and Ranging and camera), additional depth information can be utilized to better understand the images in 2-D and extend the image-level knowledge to the real world. However, seldom of these 3-D-aware techniques aim to place correct 3-D BBs around detected objects. To achieve better bounding results, multiview representation [181] and 3-D proposal network [228] may provide some guidelines to encode depth information with the aid of inertial sensors (accelerometer and gyrometer) [229].

2) *Video Object Detection:* Temporal information across different frames plays an important role in understanding the behaviors of different objects. However, the accuracy suffers from degenerated object appearances (e.g., motion blur and video defocus) in videos and the network is usually not trained end to end. To this end, spatiotemporal tubelets [230], optical flow [199], and LSTM [107] should be considered to fundamentally model object associations between consecutive frames.

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 that handle different subproblems, 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

[1] P. F. Felzenszwalb *et al.*, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[2] 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, Jan. 1998.

[3] C. Wojek *et al.*, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, Apr. 2012.

[4] 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, Jun. 1996.

[5] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM MM*, 2014, pp. 675–678.

[6] A. Krizhevsky *et al.*, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[7] Z. Cao *et al.*, "Realtime multi-person 2D pose estimation using part affinity fields," in *Proc. CVPR*, 2017, pp. 1302–1310.

[8] Z. Yang and R. Nevatia, "A multi-scale cascade fully convolutional network face detector," in *Proc. ICPR*, 2016, pp. 633–638.

[9] C. Chen *et al.*, "DeepDriving: Learning affordance for direct perception in autonomous driving," in *Proc. ICCV*, 2015, pp. 2722–2730.

[10] X. Chen *et al.*, "Multi-view 3D object detection network for autonomous driving," in *Proc. CVPR*, 2017, pp. 6526–6534.

[11] A. Dundar *et al.*, "Embedded streaming deep neural networks accelerator with applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 7, pp. 1572–1583, Jul. 2017.

[12] R. J. Cintra *et al.*, "Low-complexity approximate convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5981–5992, 2018.

[13] S. H. Khan *et al.*, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3573–3587, Aug. 2018.

[14] A. Stuhlsatz *et al.*, "Feature extraction with deep neural networks by a generalized discriminant analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 596–608, Apr. 2012.

[15] R. Girshick *et al.*, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR*, 2014, pp. 580–587.

[16] R. Girshick, "Fast R-CNN," in *Proc. ICCV*, 2015, pp. 1440–1448.

[17] S. Ren *et al.*, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. NIPS*, 2015, pp. 91–99.

[18] J. Redmon *et al.*, "You only look once: Unified, real-time object detection," in *Proc. CVPR*, 2016, pp. 779–788.

[19] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. CVPR*, 2005, pp. 886–893.

[21] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in *Proc. ICIP*, 2002, p. 1.

[22] C. Cortes and V. Vapnik, "Support vector machine," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[23] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[24] P. F. Felzenszwalb *et al.*, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[25] M. Everingham *et al.*, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2008.

[26] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[27] Y. LeCun *et al.*, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[28] N. Liu *et al.*, "Predicting eye fixations using convolutional neural networks," in *Proc. CVPR*, 2015, pp. 362–370.

[29] E. Vig *et al.*, "Large-scale optimization of hierarchical features for saliency prediction in natural images," in *Proc. CVPR*, 2014, pp. 2798–2805.

[30] H. Jiang and E. Learned-Miller, "Face detection with the faster R-CNN," in *Proc. FG*, 2017, pp. 650–657.

[31] D. Chen *et al.*, "Joint cascade face detection and alignment," in *Proc. ECCV*, 2014, pp. 109–122.

[32] D. Chen *et al.*, "Supervised transformer network for efficient face detection," in *Proc. ECCV*, 2016, pp. 122–138.

[33] A. Mateus *et al.*. (2016). "Efficient and robust pedestrian detection using deep learning for human-aware navigation." [Online]. Available: https://arxiv.org/abs/1607.04441

[34] F. Yang *et al.*, "Exploit all the layers: Fast and accurate CNN object detector with scale dependent pooling and cascaded rejection classifiers," in *Proc. CVPR*, 2016, pp. 2129–2137.

[35] P. N. Druzhkov and V. D. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognit. Image Anal.*, vol. 26, no. 1, pp. 9–15, 2016.

[36] W. Pitts and W. S. McCulloch, "How we know universals the perception of auditory and visual forms," *Bull. Math. Biophys.*, vol. 9, no. 3, pp. 127–147, 1947.

[37] D. E. Rumelhart *et al.*, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[38] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[39] J. Deng *et al.*, "ImageNet: A large-scale hierarchical image database," in *Proc. CVPR*, 2009, pp. 248–255.

[40] L. Deng *et al.*, "Binary coding of speech spectrograms using a deep auto-encoder," in *Proc. INTERSPEECH*, 2010, pp. 1692–1695.

[41] G. Dahl *et al.*, "Phone recognition with the mean-covariance restricted Boltzmann machine," in *Proc. NIPS*, 2010, pp. 469–477.

[42] G. E. Hinton *et al.*. (2012). "Improving neural networks by preventing co-adaptation of feature detectors." [Online]. Available: https://arxiv.org/abs/1207.0580

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[44] P. Sermanet *et al.*. (2013). "OverFeat: Integrated recognition, localization and detection using convolutional networks." [Online]. Available: https://arxiv.org/abs/1312.6229

[45] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. CVPR*, 2015, pp. 1–9.

[46] K. Simonyan and A. Zisserman. (2014)."Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[47] K. He *et al.*, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[48] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010, pp. 807–814.

[49] M. Oquab *et al.*, "Weakly supervised object recognition with convolutional neural networks," in *Proc. NIPS*, 2014, pp. 1–10.

[50] M. Oquab *et al.*, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. CVPR*, 2014, pp. 1717–1724.

[51] F. M. Wadley, "Probit analysis: A statistical treatment of the sigmoid response curve," *Ann. Entomol. Soc. Amer.*, vol. 67, no. 4, pp. 549–553, 1947.

[52] K. Kavukcuoglu *et al.*, "Learning invariant features through topographic filter maps," in *Proc. CVPR*, 2009, pp. 1605–1612.

[53] K. Kavukcuoglu *et al.*, "Learning convolutional feature hierarchies for visual recognition," in *Proc. NIPS*, 2010, pp. 1090–1098.

[54] M. D. Zeiler *et al.*, "Deconvolutional networks," in *Proc. CVPR*, 2010, pp. 2528–2535.

[55] H. Noh *et al.*, "Learning deconvolution network for semantic segmentation," in *Proc. ICCV*, 2015, pp. 1520–1528.

[56] Z.-Q. Zhao *et al.*, "Plant leaf identification via a growing convolution neural network with progressive sample learning," in *Proc. ACCV*, 2014, pp. 348–361.

[57] A. Babenko *et al.*, "Neural codes for image retrieval," in *Proc. ECCV*, 2014, pp. 584–599.

[58] J. Wan *et al.*, "Deep learning for content-based image retrieval: A comprehensive study," in *ACM MM*, 2014, pp. 157–166.

[59] D. Tomè *et al.*, "Deep convolutional neural networks for pedestrian detection," *Signal Process., Image Commun.*, vol. 47, pp. 482–489, Sep. 2016.

[60] Y. Xiang *et al.*, "Subcategory-aware convolutional neural networks for object proposals and detection," in *Proc. WACV*, 2017, pp. 924–933.

[61] Z.-Q. Zhao *et al.*, "Pedestrian detection based on fast R-CNN and batch normalization," in *Proc. ICIC*, 2017, pp. 735–746.

[62] J. Ngiam *et al.*, "Multimodal deep learning," in *Proc. ICML*, 2011, pp. 689–696.

[63] Z. Wu *et al.*, "Modeling spatial-temporal clues in a hybrid deep learning framework for video classification," in *Proc. ACM MM*, 2015, pp. 461–470.

[64] K. He *et al.*, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[65] J. Dai *et al.*, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. NIPS*, 2016, pp. 379–387.

[66] T.-Y. Lin *et al.*, "Feature pyramid networks for object detection," in *Proc. CVPR*, 2017, pp. 936–944.

[67] K. He *et al.*, "Mask R-CNN," in *Proc. ICCV*, 2017, pp. 2980–2988.

[68] D. Erhan *et al.*, "Scalable object detection using deep neural networks," in *Proc. CVPR*, 2014, pp. 2155–2162.

[69] D. Yoo *et al.*, "AttentionNet: Aggregating weak directions for accurate object detection," in *Proc. CVPR*, 2015, pp. 2659–2667.

[70] M. Najibi *et al.*, "G-CNN: An iterative grid based object detector," in *Proc. CVPR*, 2016, pp. 2369–2377.

[71] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. ECCV*, 2016, pp. 21–37.

[72] J. Redmon and A. Farhadi. (2016). "YOLO9000: Better, faster, stronger." [Online]. Available: https://arxiv.org/abs/1612.08242

[73] C.-Y. Fu *et al.*. (2017). "DSSD : Deconvolutional single shot detector." [Online]. Available: https://arxiv.org/abs/1701.06659

[74] Z. Shen *et al.*, "DSOD: Learning deeply supervised object detectors from scratch," in *Proc. ICCV*, 2017, p. 7.

[75] G. E. Hinton *et al.*, "Transforming auto-encoders," in *Proc. ICANN*, 2011, pp. 44–51.

[76] G. W. Taylor *et al.*, "Learning invariance through imitation," in *Proc. CVPR*, 2011, pp. 2729–2736.

[77] X. Ren and D. Ramanan, "Histograms of sparse codes for object detection," in *Proc. CVPR*, 2013, pp. 3246–3253.

[78] J. R. R. Uijlings *et al.*, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.

[79] P. Sermanet *et al.*, "Pedestrian detection with unsupervised multi-stage feature learning," in *Proc. CVPR*, 2013, pp. 3626–3633.

[80] P. Krähenbühl and V. Koltun, "Geodesic object proposals," in *Proc. ECCV*, 2014, pp. 725–739.

[81] P. Arbeláez *et al.*, "Multiscale combinatorial grouping," in *Proc. CVPR*, 2014, pp. 328–335.

[82] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. ECCV*, 2014, pp. 391–405.

[83] W. Kuo *et al.*, "Deepbox: Learning objectness with convolutional networks," in *Proc. ICCV*, 2015, pp. 2479–2487.

[84] P. O. Pinheiro *et al.*, "Learning to refine object segments," in *Proc. ECCV*, 2016, pp. 75–91.

[85] Y. Zhang *et al.*, "Improving object detection with deep convolutional networks via Bayesian optimization and structured prediction," in *Proc. CVPR*, 2015, pp. 249–258.

[86] S. Gupta *et al.*, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. ECCV*, 2014, pp. 345–360.

[87] W. Ouyang *et al.*, "DeepID-Net: Deformable deep convolutional neural networks for object detection," in *Proc. CVPR*, 2015, pp. 2403–2412.

[88] K. Lenc and A. Vedaldi. (2015). "R-CNN minus R." [Online]. Available: https://arxiv.org/abs/1506.06981

[89] S. Lazebnik *et al.*, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. CVPR*, 2006, pp. 2169–2178.

[90] F. Perronnin *et al.*, "Improving the Fisher kernel for large-scale image classification," in *Proc. ECCV*, 2010, pp. 143–156.

[91] J. Xue *et al.*, "Restructuring of deep neural network acoustic models with singular value decomposition," in *Proc. INTERSPEECH*, 2013, pp. 2365–2369.

[92] S. Ren *et al.*, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[93] C. Szegedy *et al.*, "Rethinking the inception architecture for computer vision," in *Proc. CVPR*, 2016, pp. 2818–2826.

[94] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. ECCV*, 2014, pp. 740–755.

[95] S. Bell *et al.*, "Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks," in *Proc. CVPR*, 2016, pp. 2874–2883.

[96] A. Arnab and P. H. S. Torr, "Pixelwise instance segmentation with a dynamically instantiated network," in *Proc. CVPR*, 2017, pp. 879–888.

[97] J. Dai *et al.*, "Instance-aware semantic segmentation via multi-task network cascades," in *Proc. CVPR*, 2016, pp. 3150–3158.

[98] Y. Li *et al.*, "Fully convolutional instance-aware semantic segmentation," in *Proc. CVPR*, 2017, pp. 4438–4446.

[99] M. Jaderberg *et al.*, "Spatial transformer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2017–2025.

[100] S. Brahmbhatt *et al.*, "StuffNet: Using 'Stuff' to improve object detection," in *Proc. WACV*, 2017, pp. 934–943.

[101] T. Kong *et al.*, "HyperNet: Towards accurate region proposal generation and joint object detection," in *Proc. CVPR*, 2016, pp. 845–853.

[102] A. Pentina *et al.*, "Curriculum learning of multiple tasks," in *Proc. CVPR*, 2015, pp. 5492–5500.

[103] J. Yim *et al.*, "Rotating your face using multi-task deep neural network," in *Proc. CVPR*, 2015, pp. 676–684.

[104] J. Li *et al.*. (2016). "Multi-stage object detection with group recursive learning." [Online]. Available: https://arxiv.org/abs/1608.05159

[105] Z. Cai *et al.*, "A unified multi-scale deep convolutional neural network for fast object detection," in *Proc. ECCV*, 2016, pp. 354–370.

[106] Y. Zhu *et al.*, "segDeepM: Exploiting segmentation and context in deep neural networks for object detection," in *Proc. CVPR*, 2015, pp. 4703–4711.

[107] W. Byeon *et al.*, "Scene labeling with LSTM recurrent neural networks," in *Proc. CVPR*, 2015, pp. 3547–3555.

[108] B. Moysset *et al.*. (2016). "Learning to detect and localize many objects from few examples." [Online]. Available: https://arxiv.org/abs/1611.05664

[109] X. Zeng *et al.*, "Gated bi-directional CNN for object detection," in *Proc. ECCV*, 2016, pp. 354–369.

[110] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware CNN model," in *Proc. CVPR*, 2015, pp. 1134–1142.

[111] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[112] S. Zagoruyko *et al.* (2016). "A multiPath network for object detection." [Online]. Available: https://arxiv.org/abs/1604.02135

[113] A. Shrivastava *et al.*, "Training region-based object detectors with online hard example mining," in *Proc. CVPR*, 2016, pp. 761–769.

[114] S. Ren *et al.*, "Object detection networks on convolutional feature maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1476–1481, Jul. 2017.

[115] W. Ouyang *et al.*, "Factors in finetuning deep model for object detection with long-tail distribution," in *Proc. CVPR*, 2016, pp. 864–873.

[116] S. Hong *et al.*. (2016). "PVANet: Lightweight deep neural networks for real-time object detection." [Online]. Available: https://arxiv.org/abs/1611.08588

[117] W. Shang *et al.*, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *Proc. ICML*, 2016, pp. 1–9.

[118] C. Szegedy *et al.*, "Deep neural networks for object detection," in *Proc. NIPS*, 2013, pp. 1–9.

[119] P. O. Pinheiro *et al.*, "Learning to segment object candidates," in *Proc. NIPS*, 2015, pp. 1990–1998.

[120] C. Szegedy *et al.*. (2014). "Scalable, high-quality object detection." [Online]. Available: https://arxiv.org/abs/1412.1441

[121] M. Everingham *et al.*. (2011). *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results (2012)*. [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html

[122] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. ECCV*, 2014, pp. 818–833.

[123] S. Xie *et al.*, "Aggregated residual transformations for deep neural networks," in *Proc. CVPR*, 2017, pp. 5987–5995.

[124] "J. Dai *et al.* (2017). "Deformable convolutional networks." [Online]. Available: https://arxiv.org/abs/1703.06211

[125] C. Rother *et al.*, "AutoCollage," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 847–852, 2006.

[126] C. Jung and C. Kim, "A unified spectral-domain approach for saliency detection and its application to automatic object segmentation," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1272–1283, Mar. 2012.

[127] W.-C. Tu *et al.*, "Real-time salient object detection with a minimum spanning tree," in *Proc. CVPR*, 2016, pp. 2334–2342.

[128] J. Yang and M.-H. Yang, "Top-down visual saliency via joint CRF and dictionary learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 3, pp. 576–588, Mar. 2017.

[129] P. L. Rosin, "A simple method for detecting salient regions," *Pattern Recognit.*, vol. 42, no. 11, pp. 2363–2371, Nov. 2009.

[130] T. Liu *et al.*, "Learning to detect a salient object," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 353–367, Feb. 2011.

[131] J. Long *et al.*, "Fully convolutional networks for semantic segmentation," in *Proc. CVPR*, 2015, pp. 3431–3440.

[132] D. Gao *et al.*, "Discriminant saliency, the detection of suspicious coincidences, and applications to visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 6, pp. 989–1005, Jun. 2009.

[133] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. ICCV*, 2015, pp. 1395–1403.

[134] M. Kümmerer *et al.*. (2014). "Deep gaze I: Boosting saliency prediction with feature maps trained on ImageNet." [Online]. Available: https://arxiv.org/abs/1411.1045

[135] X. Huang *et al.*, "SALICON: Reducing the semantic gap in saliency prediction by adapting deep neural networks," in *Proc. ICCV*, 2015, pp. 262–270.

[136] L. Wang *et al.*, "Deep networks for saliency detection via local estimation and global search," in *Proc. CVPR*, 2015, pp. 3183–3192.

[137] H. Cholakkal *et al.*. (2016). "Backtracking spatial pyramid pooling (SPP)-based image classifier for weakly supervised top-down salient object detection." [Online]. Available: https://arxiv.org/abs/1611.05345

[138] R. Zhao *et al.*, "Saliency detection by multi-context deep learning," in *Proc. CVPR*, 2015, pp. 1265–1274.

[139] C. Bak *et al.*. (2016). "Spatio-temporal saliency networks for dynamic saliency prediction." [Online]. Available: https://arxiv.org/abs/1607.04730

[140] S. He *et al.*, "SuperCNN: A superpixelwise convolutional neural network for salient object detection," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 330–344, 2015.

[141] X. Li *et al.*, "DeepSaliency: Multi-task deep neural network model for salient object detection," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3919–3930, Aug. 2016.

[142] Y. Tang and X. Wu, "Saliency detection via combining region-level and pixel-level predictions with CNNs," in *Proc. ECCV*, 2016, pp. 809–825.

[143] G. Li and Y. Yu, "Deep contrast learning for salient object detection," in *Proc. CVPR*, 2016, pp. 478–487.

[144] X. Wang *et al.*. (2016). "Edge preserving and multi-scale contextual neural network for salient object detection." [Online]. Available: https://arxiv.org/abs/1608.08029

[145] M. Cornia *et al.*, "A deep multi-level network for saliency prediction," in *Proc. ICPR*, 2016, pp. 3488–3493.

[146] G. Li and Y. Yu, "Visual saliency detection based on multiscale deep CNN features," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5012–5024, Nov. 2016.

[147] J. Pan *et al.*, "Shallow and deep convolutional networks for saliency prediction," in *Proc. CVPR*, 2016, pp. 598–606.

[148] J. Kuen *et al.*, "Recurrent attentional networks for saliency detection," in *Proc. CVPR*, 2016, pp. 3668–3677.

[149] Y. Tang *et al.*, "Deeply-supervised recurrent convolutional neural network for saliency detection," in *Proc. ACM MM*, 2016, pp. 397–401.

[150] X. Li *et al.*, "Contextual hypergraph modeling for salient object detection," in *Proc. ICCV*, 2013, pp. 3328–3335.

[151] M.-M. Cheng *et al.*, "Global contrast based salient region detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 569–582, Mar. 2015.

[152] H. Jiang *et al.*, "Salient object detection: A discriminative regional feature integration approach," in *Proc. CVPR*, 2013, pp. 2083–2090.

[153] G. Lee *et al.*, "Deep saliency with encoded low level distance map and high level features," in *Proc. CVPR*, 2016, pp. 660–668.

[154] Z. Luo *et al.*, "Non-local deep features for salient object detection," in *Proc. CVPR*, 2017, pp. 6593–6601.

[155] Q. Hou *et al.*. (2016). "Deeply supervised salient object detection with short connections." [Online]. Available: https://arxiv.org/abs/1611.04849

[156] Q. Yan *et al.*, "Hierarchical saliency detection," in *Proc. CVPR*, 2013, pp. 1155–1162.

[157] Y. Li *et al.*, "The secrets of salient object segmentation," in *Proc. CVPR*, 2014, pp. 280–287.

[158] V. Movahedi and J. H. Elder, "Design and perceptual validation of performance measures for salient object segmentation," in *Proc. CVPRW*, 2010, pp. 49–56.

[159] A. Borji *et al.*, "Salient object detection: A benchmark," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5706–5722, Dec. 2015.

[160] C. Peng *et al.*, "Graphical representation for heterogeneous face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 2, pp. 301–312, Feb. 2017.

[161] C. Peng *et al.*, "Face recognition from multiple stylistic sketches: Scenarios, datasets, and evaluation," in *Proc. ECCV*, 2016, pp. 3–18.

[162] X. Gao *et al.*, "Face sketch–photo synthesis and retrieval using sparse representation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 8, pp. 1213–1226, Aug. 2012.

[163] N. Wang *et al.*, "A comprehensive survey to face hallucination," *Int. J. Comput. Vis.*, vol. 106, no. 1, pp. 9–30, 2014.

[164] C. Peng *et al.*, "Multiple representations-based face sketch–photo synthesis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2201–2215, Nov. 2016.

[165] A. Majumder *et al.*, "Automatic facial expression recognition system using deep network-based data fusion," *IEEE Trans. Cybern.*, vol. 48, no. 1, pp. 103–114, Jan. 2018.

[166] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2004.

[167] J. Yu *et al.*, "Unitbox: An advanced object detection network," in *Proc. ACM MM*, 2016, pp. 516–520.

[168] S. S. Farfade *et al.*, "Multi-view face detection using deep convolutional neural networks," in *Proc. ICMR*, 2015, pp. 643–650.

[169] S. Yang *et al.*, "From facial parts responses to face detection: A deep learning approach," in *Proc. ICCV*, 2015, pp. 3676–3684.

[170] S. Yang *et al.*, "Face detection through scale-friendly deep convolutional networks," in *Proc. CVPR*, 2017, pp. 1–12.

[171] Z. Hao *et al.*, "Scale-aware face detection," in *Proc. CVPR*, 2017, pp. 1913–1922.

[172] H. Wang *et al.*. (2017). "Face R-CNN." [Online]. Available: https://arxiv.org/abs/1706.01061

[173] X. Sun *et al.*. (2017). "Face detection using deep learning: An improved faster RCNN approach." [Online]. Available: https://arxiv.org/abs/1701.08289

[174] L. Huang *et al.*. (2015). "DenseBox: Unifying landmark localization with end to end object detection." [Online]. Available: https://arxiv.org/abs/1509.04874

[175] Y. Li *et al.*, "face detection with end-to-end integration of a ConvNet and a 3D model," in *Proc. ECCV*, 2016, pp. 420–436.

[176] K. Zhang *et al.*, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, Oct. 2016.

[177] I. A. Kalinovsky and V. G. Spitsyn, "Compact convolutional neural network cascade for face detection," in *Proc. CEUR Workshop*, 2016, pp. 375–387.

[178] H. Qin *et al.*, "Joint training of cascaded CNN for face detection," in *Proc. CVPR*, 2016, pp. 3456–3465.

[179] V. Jain and E. Learned-Miller, "FDDB: A benchmark for face detection in unconstrained settings," Univ. Massachusetts, Amherst, Amherst, MA, USA, Tech. Rep. UM-CS-2010-009, 2010.

[180] H. Li *et al.*, "A convolutional neural network cascade for face detection," in *Proc. CVPR*, 2015, pp. 5325–5334.

[181] B. Yang *et al.*, "Aggregate channel features for multi-view face detection," in *Proc. IJCB*, 2014, pp. 1–8.

[182] N. Markuš *et al.*. (2013). "Object detection with pixel intensity comparisons organized in decision trees." [Online]. Available: https://arxiv.org/abs/1305.4537

[183] M. Mathias *et al.*, "Face detection without bells and whistles," in *Proc. ECCV*, 2014.

[184] J. Li and Y. Zhang, "Learning surf cascade for fast and accurate object detection," in *Proc. CVPR*, 2013.

[185] S. Liao *et al.*, "A fast and accurate unconstrained face detector," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 211–223, Feb. 2016.

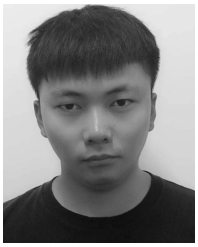
[186] B. Yang *et al.*, "Convolutional channel features," in *Proc. ICCV*, 2015, pp. 82–90.
[187] R. Ranjan *et al.*. (2016). "HyperFace: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition." [Online]. Available: https://arxiv.org/abs/1603.01249
[188] P. Hu and D. Ramanan, "Finding tiny faces," in *Proc. CVPR*, 2017, pp. 1522–1530.
[189] Z. Jiang and D. Q. Huynh, "Multiple pedestrian tracking from monocular videos in an interacting multiple model framework," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1361–1375, Mar. 2018.
[190] D. M. Gavrila and S. Munder, "Multi-cue pedestrian detection and tracking from a moving vehicle," *Int. J. Comput. Vis.*, vol. 73, no. 1, pp. 41–59, Jun. 2007.
[191] S. Xu *et al.*, "Jointly attentive spatial-temporal pooling networks for video-based person re-identification," in *Proc. ICCV*, 2017, pp. 4743–4752.
[192] Z. Liu *et al.*, "Stepwise metric promotion for unsupervised video person re-identification," in *Proc. ICCV*, 2017, pp. 2448–2457.
[193] A. Khan *et al.*, "Cooperative robots to observe moving targets: Review," *IEEE Trans. Cybern.*, vol. 48, no. 1, pp. 187–198, Jan. 2018.
[194] A. Geiger *et al.*, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.
[195] Z. Cai *et al.*, "Learning complexity-aware cascades for deep pedestrian detection," in *Proc. ICCV*, 2015, pp. 3361–3369.
[196] Y. Tian *et al.*, "Deep learning strong parts for pedestrian detection," in *Proc. CVPR*, 2015, pp. 1904–1912.
[197] P. Dollár *et al.*, "Fast feature pyramids for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1532–1545, Aug. 2014.
[198] S. Zhang *et al.*, "Filtered channel features for pedestrian detection," in *Proc. CVPR*, 2015, pp. 1751–1760.
[199] S. Paisitkriangkrai *et al.*, "Pedestrian detection with spatially pooled features and structured ensemble learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 6, pp. 1243–1257, Jun. 2016.
[200] L. Lin *et al.*, "Discriminatively trained And-Or graph models for object shape detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 5, pp. 959–972, May 2015.
[201] M. Mathias *et al.*, "Handling occlusions with Franken-classifiers," in *Proc. ICCV*, 2013, pp. 1505–1512.
[202] S. Tang *et al.*, "Detection and tracking of occluded people," *Int. J. Comput. Vis.*, vol. 110, no. 1, pp. 58–69, 2014.
[203] L. Zhang *et al.*, "Is faster R-CNN doing well for pedestrian detection?" in *Proc. ECCV*, 2016, pp. 443–457.
[204] Y. Tian *et al.*, "Deep learning strong parts for pedestrian detection," in *Proc. ICCV*, 2015.
[205] J. Liu *et al.*. (2016). "Multispectral deep neural networks for pedestrian detection." [Online]. Available: https://arxiv.org/abs/1611.02644
[206] Y. Tian *et al.*, "Pedestrian detection aided by deep learning semantic tasks," in *Proc. CVPR*, 2015, pp. 5079–5087.
[207] X. Du *et al.*, "Fused DNN: A deep neural network fusion approach to fast and robust pedestrian detection," in *Proc. WACV*, 2017, pp. 953–961.
[208] Q. Hu *et al.*, "Pushing the limits of deep CNNs for pedestrian detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1358–1368, Jun. 2018.
[209] D. Tomé *et al.*, "Reduced memory region based deep convolutional neural network detection," in *Proc. ICCE*, Berlin, Germany, 2016, pp. 15–19.
[210] J. Hosang *et al.*, "Taking a deeper look at pedestrians," in *Proc. CVPR*, 2015, pp. 4073–4082.
[211] J. Li *et al.*. (2015). "Scale-aware fast R-CNN for pedestrian detection." [Online]. Available: https://arxiv.org/abs/1510.08160
[212] Y. Gao *et al.*, "Visual-textual joint relevance learning for tag-based social image search," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 363–376, Jan. 2013.
[213] T. Kong *et al.*, "RON: Reverse connection with objectness prior networks for object detection," in *Proc. CVPR*, 2017, pp. 5244–5252.
[214] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. NIPS*, 2014, pp. 2672–2680.
[215] Y. Fang *et al.*, "Object detection meets knowledge graphs," in *Proc. IJCAI*, 2017, pp. 1661–1667.
[216] S. Welleck *et al.*, "Saliency-based sequential image attention with multiset prediction," in *Proc. NIPS*, 2017, pp. 5173–5183.
[217] S. Azadi *et al.*, "Learning detection with diverse proposals," in *Proc. CVPR*, 2017, pp. 7369–7377.
[218] S. Sukhbaatar *et al.*, "End-to-end memory networks," in *Proc. NIPS*, 2015, pp. 2440–2448.
[219] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in *Proc. NIPS*, 2017, pp. 6967–6976.
[220] B. Yang *et al.*, "CRAFT objects from images," in *Proc. CVPR*, 2016, pp. 6043–6051.
[221] I. Croitoru *et al.*, "Unsupervised learning from video to detect foreground objects in single images," in *Proc. ICCV*, 2017, pp. 4345–4353.
[222] C. Wang *et al.*, "Weakly supervised object localization with latent category learning," in *Proc. ECCV*, 2014, pp. 431–445.
[223] D. P. Papadopoulos *et al.*, "Training object class detectors with click supervision," in *Proc. CVPR*, 2017, pp. 180–189.
[224] J. Huang *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proc. CVPR*, 2017, pp. 3296–3297.
[225] Q. Li *et al.*, "Mimicking very efficient network for object detection," in *Proc. CVPR*, 2017, pp. 7341–7349.
[226] G. Hinton *et al.*, "Distilling the knowledge in a neural network," *Comput. Sci.*, vol. 14, no. 7, pp. 38–39, 2015.
[227] A. Romero *et al.*, "FitNets: Hints for thin deep nets," in *Proc. ICLR*, 2015, pp. 1–13.
[228] X. Chen *et al.*, "3D object proposals for accurate object class detection," in *Proc. NIPS*, 2015, pp. 424–432.
[229] J. Dong *et al.*, "Visual-inertial-semantic scene representation for 3D object detection," in *Proc. CVPR*, 2017, pp. 960–970.
[230] K. Kang *et al.*, "Object detection in videos with tubelet proposal networks," in *Proc. CVPR*, 2017, pp. 889–897.

Zhong-Qiu Zhao (M'10) received the Ph.D. degree in pattern recognition and intelligent system from the University of Science and Technology of China, Hefei, China, in 2007.

From 2008 to 2009, he held a post-doctoral position in image processing with the CNRS UMR6168 Lab Sciences de Information et des Systèmes, La Garde, France. From 2013 to 2014, he was a Research Fellow in image processing with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. He is currently a Professor with the Hefei University of Technology, Hefei. His current research interests include pattern recognition, image processing, and computer vision.

Peng Zheng received the bachelor's degree from the Hefei University of Technology, Hefei, China, in 2010, where he is currently pursuing the Ph.D. degree.

His current research interests include pattern recognition, image processing, and computer vision.



Shou-Tao Xu is currently pursuing the master's degree with the Hefei University of Technology, Hefei, China.

His current research interests include pattern recognition, image processing, deep learning, and computer vision.

Xindong Wu (F'11) received the Ph.D. degree in artificial intelligence from The University of Edinburgh, Edinburgh, U.K.

He is currently an Alfred and Helen Lamson Endowed Professor of computer science with the University of Louisiana at Lafayette, Lafayette, LA, USA. His current research interests include data mining, knowledge-based systems, and Web information exploration.

Dr. Wu is a Fellow of the AAAS. He is the Steering Committee Chair of the IEEE International Conference on Data Mining. He served as the Editor-in-Chief for the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING (IEEE Computer Society) between 2005 and 2008.