International Journal of Scientific Research in Engineering and Management (IJSREM)

Volume: 04 Issue: 07 | July -2020 ISSN: 2582-3930

Object Detection and Classification between Crawling: A Review

Priyank Mishra, Galgotias University final semester (B.tech C.S.E.)
Under guidance of –Mr. Anurag Singh

Abstract:-An optical camera-based intrusion arrangement (Light Intrusion DeTection systEm named as acronym LITE) for an outside setting was recently developed by a superset of the authors. The system classified between human and animal images captured during a side-view manner supported the peak . supported the system and algorithm design, most likely human-crawl would be classified as animal by the LITE. during this paper, classification between human-crawl and animal is addressed. additionally to the present work, classification of person with weapon versus person with vehicle is additionally addressed (referred as person with Crawling) to supply more information about the sort of intrusions. A Convolutional Neural Network (CNN) based approach is employed to unravel the above stated two problems. within the case of "person with Crawling" classification, a study of various CNN architectures was administered and analysis like that's presented. just in case of human crawl vs animal movement, performance results like only the simplest architecture model is provided among the various tried models. Further on, additional insights are provided about the classification using the eye heat maps and t-SNE plots. The test classification accuracies for human-crawl vs animal and person with Crawling classification on the recorded data are on the brink of 95.65% and 90%, respectively. The LITE, having the Odroid C2 (OC2) Single-Board Computer (SBC) with CNN-based classification algorithm for human-crawl versus animal task ported thereon , was deployed in an outside setting for a realtime deployment. It provided a classification accuracy on the brink of 92%. Traditional Crawling detection methods are built on handcrafted features and shallow trainable architectures. Their performance easily stagnates by constructing complex ensembles which combine multiple low-level image features with high-level context from Crawling detectors and scene classifiers. With the rapid development in deep learning, more powerful tool, which are ready to learn semantic, high-level, deeper features, are introduced to deal with the issues existing in traditional architectures. These models behave differently in specification , training strategy and optimization function, etc. during this paper, we offer a review on deep learning based Crawling detection frameworks. Our review begins with a quick introduction on the history of deep learning and its representative tool, namely Convolutional Neural Network (CNN). Then we specialize in typical generic Crawling detection architectures along side some modifications and useful tricks to enhance detection performance further. As distinct specific detection tasks exhibit different characteristics, we also briefly survey several specific tasks, including salient Crawling detection, crawl detection and crawling object detection. Experimental analyses also provided to match various methods and draw some meaningful conclusions. Finally, several promising directions and tasks are provided to function guidelines for future add both Crawling detection and relevant neural network based learning systems.

Index Terms—deep learning, Crawling detection, neural network

I. INTRODUCTION O gain an entire image understanding, we should always not only consider classifying different images, but also attempt to precisely estimate the concepts and locations of Crawlings contained in each image. This task is referred as Crawling detection [1][S1], which usually consists of various subtasks like crawl detection [2][S2], crawling object detection [3][S2] and skeleton detection [4][S3]. together of the elemental computer vision problems, Crawling detection is in a position to supply valuable information for semantic understanding of images and videos, and is said to several applications, including image classification [5], [6], human behavior analysis [7][S4], crawl 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 can even have great impacts on Crawling detection techniques which may be considered as learning systems. [11]–[14][S6]. However, thanks to large variations in viewpoints, poses, occlusions and lighting conditions, it's difficult to perfectly accomplish Crawling detection with an additiona lCrawling localization task. such a lot attention has been interested in this field in recent years [15]–[18].

The problem definition of Crawling detection is to work out where Crawlings are located during a given image (Crawling localization) and which category each Crawling belongs to (Crawling classification). therefore the pipeline of traditional Crawling detection models are often mainly divided into

Volume: 04 Issue: 07 | July -2020 ISSN: 2582-3930

three stages: informative region selection, feature extraction and classification.

Informative region selection. As different Crawlings may appear in any positions of the image and have different aspect ratios or sizes, it's a natural option to scan the entire image with a multi-scale window . Although this exhaustive strategy can determine all possible positions of the Crawlings, its shortcomings also are obvious. thanks to an outsized number of candidate windows, it's computationally expensive and produces too many redundant windows. However, if only a hard and fast number of window templates are applied, unsatisfactory regions could also be produced.

Feature extraction. to acknowledge different Crawlings, we'd like to extract visual features which may provide a semantic and robust representation. SIFT [19], HOG [20] and Haar-like [21] features are the representative ones. this is often thanks to the very fact that these features can produce representations related to complex cells in human brain [19]. However, thanks to the range of appearances, illumination conditions and backgrounds, it's difficult to manually design a strong feature descriptor to perfectly describe all types of Crawlings.

Classification. Besides, a classifier is required to differentiate a target Crawling from all the opposite categories and to form 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 may be a flexible model by combining Crawling parts with deformation cost to handle severe deformations. In DPM, with the help of a graphical model, carefully designed low-level features and kinematically inspired part decompositions are combined. And discriminative learning of graphical models allows for building high-precision part-based models for a spread of Crawling classes.

Based on these discriminant local feature descriptors and shallow learnable architectures, state of the art results are obtained on PASCAL VOC Crawling detection competition [25] and real-time embedded systems are obtained with a coffee 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 with a sliding window strategy is redundant, inefficient and inaccurate. 2) The semantic gap cannot be

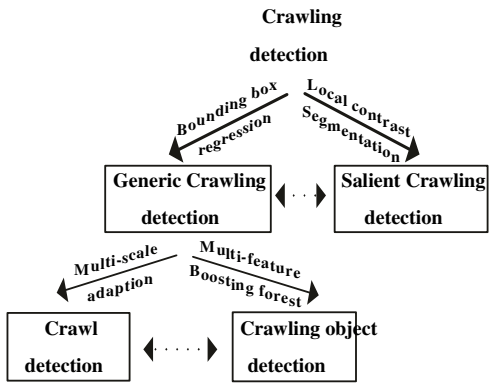


Fig. 1. The application domains of Crawling detection.

bridge by **the mixture** of manually engineered low level descriptors and discriminatively-trained shallow model. Greeting towards the emergency of Deep Neural Networks (DNNs) [6][S7], a more significant gain is obtained with the introduction of Regions with CNN features (R-CNN) [15]. DNNs, or **the foremost** representative CNNs, act **during a** quite different way from traditional approaches. **they need** deeper architectures with the capacity **to find out** more complex features than the shallow ones. Also the expressivity and robust training algorithms allow **to find out** informative Crawling representations without **the necessity to style** features manually [26].

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

In this paper, **a scientific** review is provided to summarise representative models and their different characteristics in several application domains, including generic Crawling detection [15], [16], [18], salient Crawling detection [27], [28], crawl detection [29]–[31] and crawl detection [32], [33]. Their relationships are depicted in Figure 1. **Supported** basic CNN architectures, generic Crawling detection is achieved with bounding box regression, while salient Crawling detection is accomplished with local contrast enhancement and pixel-level segmentation. Crawling detection and crawling detection are closely **associated with** generic Crawling detection and mainly accomplished with multi-scale adaption and multi-feature fusion/boosting forest, respectively. The dotted lines indicate that the corresponding domains are **related to one another** under certain conditions. It should be noticed that the covered domains are diversified. Crawling object and crawl images have regular structures, while general Crawlings and scene by the combination of

manually engineered low-level descriptors and discriminatively-trained shallow models.

Thanks to the emergency of Deep Neural Networks (DNNs) [6][S7], a more significant gain is obtained with the introduction of Regions with CNN features (R-CNN) [15]. DNNs, or **the foremost** representative CNNs, act **during a** quite different way from traditional approaches. **they need** deeper architectures with the capacity **to find out** more complex features than the shallow ones. Also the expressivity and robust training algorithms allow **to find out** informative Crawling representations without **the necessity to style** features manually [26].

Since the proposal of R-CNN, **an excellent** deal of improved models **are** suggested, including Fast R-CNN which jointly optimizes classification and bounding box regression tasks [16], Faster R-CNN which takes **a**

further subnetwork to get region proposals [18] and YOLO which accomplishes Crawling detection via a fixed-grid regression [17]. All of them bring different degrees of detection performance improvements over the first R-CNN and make real-time and accurate Crawling detection become more achievable.

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

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

The rest of this paper is organized as follows.

In Section2, a quick introduction on the history of deep learning and therefore the basic architecture of CNN is provided. Generic Crawling detection architectures are presented in Section 3. Then reviews of CNN applied in several specific tasks, including salient Crawling detection, crawl detection and crawling object detection, are exhibited in Section 4-6, respectively. Several promising future directions are proposed in Section 7. At last, some concluding remarks are presented in Section 8.

II. a quick OVERVIEW OF DEEP LEARNING

Prior to overview on deep learning based Crawling detection approaches, we provide a review on the history of deep learning in conjunction with an introduction on the essential architecture and advantages of CNN.

A. The History: Birth, Decline and Prosperity

Deep models are often mentioned as neural networks with deep structures. The history of neural networks can go back to 1940s [35], and therefore the original intention was to simulate the human brain system to unravel general learning problems during a principled way. it had been popular in 1980s and 1990s with the proposal of back-propagation algorithm by Hinton et al. [36]. However, thanks to the overfitting of coaching , lack of huge scale training data, limited computation power and insignificance in performance compared with other machine learning tools, neural networks fell out of fashion in early 2000s.

Deep learning has become popular since 2006 [37][S7] with an opportunity through in speech recognition [38]. The recovery of deep learning are often attributed to the subsequent factors.

•As ImageNet [39], to completely exhibit its very large learningThe emergence of huge scale annotated training data, such

capacity;

•Systems, like GPU clusters;Fast development of high performance parallel computing

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

What prompts deep learning to possess an enormous impact on the whole academic community? it's getting to owe to the contribution of Hinton's group, whose continuous efforts have demonstrated that deep learning would bring a revolutionary breakthrough on grand challenges rather than just obvious improvements on small datasets. Their success results from training an outsized CNN on 1.2 million labeled images along side a couple of techniques [6] (e.g., ReLU operation [48] and 'dropout' regularization).

B. Architecture and Advantages of CNN

CNN is that the most representative model of deep learning [26]. A typical CNN architecture, which is mentioned as VGG16, are often found in Fig. S1. Each layer of CNN is understood as a feature map. The feature map of the input layer may be a 3D matrix of pixel intensities for various color channels (e.g. RGB). The feature map of any internal layer is an induced multi-channel image, whose 'pixel' are often viewed as a selected feature. Every neuron is connected with alittle portion of adjacent neurons from the previous layer (receptive field). differing types of transformations [6], [49], [50] are often conducted on feature maps, like 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 get final responses. Pooling operation, like max pooling, average pooling, L2-pooling and native contrast normalization [52], summaries the responses of a receptive field into one value to supply more robust feature descriptions.

With an interleave between convolution and pooling, an initial feature hierarchy is made , which may be fine-tuned during a supervised manner by adding several fully connected (FC) layers to adapt to different visual tasks. consistent with the tasks involved, the ultimate layer with different activation functions [6] is added to urge a selected contingent probability for every output neuron. and therefore the whole network are often optimized on an Crawlingive function (e.g. mean squared error or cross-entropy loss) via the stochastic gradient descent (SGD) method. the standard VGG16 has totally 13 convolutional (conv) layers, 3 fully connected layers, 3 max-pooling layers and a softmax classification layer. The conv feature maps are produced by convoluting 3*3 filter windows, and have map resolutions are reduced with 2 stride max-pooling layers. An arbitrary test image of an equivalent size as training samples are often processed with the trained network. Re-scaling or cropping operations could also be needed if different sizes are provided [6].

The advantages of CNN against traditional methods are often summarised as follows.

•level representations from pixel to high-level semantic fea-Hierarchical feature representation, which is that the multitures learned by a hierarchical multi-stage structure [15], [53], are often learned from data automatically and, a brief introduction on the history of deep learning and therefore the basic architecture of CNN is provided. Generic Crawling detection architectures are presented in Section 3. Then reviews of CNN applied in several specific tasks, including salient Crawling detection, crawl detection and crawling object detection, are exhibited in Section 4-6, respectively. Several promising future directions are proposed in Section 7. At last, some concluding remarks are presented in Section 8.

II. A BRIEF OVERVIEW OF DEEP LEARNING

Prior to overview on deep learning based Crawling detection approaches, we offer a review on the history of deep learning along side an introduction on the essential architecture and advantages of CNN.

A. The History: Birth, Decline and Prosperity

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

Deep learning has become popular since 2006 [37][S7] with an opportunity through in speech recognition [38]. The recovery of deep learning are often attributed to the subsequent factors.

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

capacity;

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

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

What prompts deep learning to possess an enormous impact on the whole 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 instead of just obvious improvements on small datasets. Their success results from training an outsized CNN on 1.2 million labeled images together with a few techniques [6] (e.g., ReLU operation [48] and 'dropout' regularization).

III. Generic CRAWLING Detection

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

A. Region Proposal Based Framework

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

1) R-CNN: It is of significance to improve the quality of candidate bounding boxes and to take a deep architecture to extract high-level features. To solve these problems, R-CNN [15] was proposed by Ross Girshick in 2014 and obtained a mean average precision (mAP) of 53.3% with more than 30% improvement over the previous best result (DPM HSC [77]) on PASCAL VOC 2012. Figure 3 shows the flowchart of R-CNN, which can be divided into three stages as follows.

Region proposal generation. The R-CNN adopts selective search [78] to generate about 2k region proposals for each image. The selective search method relies on simple bottom-up grouping and saliency cues to provide more accurate candidate boxes of arbitrary sizes quickly and to reduce the searching space in Crawling detection [24], [39].

CNN based deep feature extraction. In this stage, each region proposal is warped or cropped into a fixed resolution and the CNN module in [6] is utilized to extract a 4096dimensional feature as the final representation. Due to large learning capacity, dominant expressive power and hierarchical structure of CNNs, a high-level, semantic and robust feature representation for each region proposal can be obtained. Classification and localization. With pre-trained categoryspecific linear SVMs for multiple classes, different region proposals are scored on a set of positive regions and background (negative) regions. The scored regions are then adjusted with

Volume: 04 Issue: 07 | July -2020 ISSN: 2582-3930

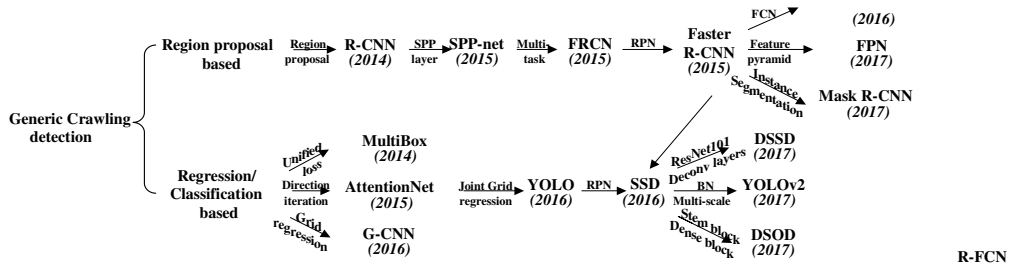
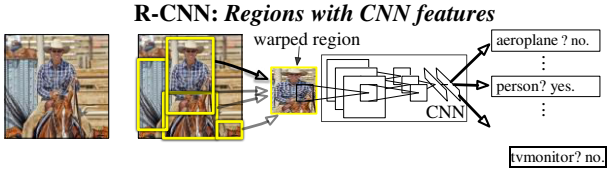


Fig. 2. Two types of frameworks: region proposal based and regression/classification based. SPP: Spatial Pyramid Pooling [64], FRCN: Faster R-CNN [16], RPN: Region Proposal Network [18], FCN: Fully Convolutional Network [65], BN: Batch Normalization [43], Deconv layers: Deconvolution layers [54].



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

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

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

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

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

- fixed-size (e.g. Due to the existence of FC layers, the CNN requires a227×227) input image, which directly leads to the re-computation of the whole CNN for each evaluated region, taking a great deal of time in the testing period.

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

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

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

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

In addition, there are some improvements to solve the problem of inaccurate localization. Zhang et al. [85] utilized a bayesian optimization based search algorithm to guide the regressions of different bounding boxes sequentially, and trained class-specific CNN classifiers with a structured loss to penalize the localization inaccuracy explicitly. Saurabh Gupta et al. improved Crawling detection for RGB-D images with semantically rich image and depth features [86], and learned a new geocentric embedding for depth images to encode each pixel. The combination of Crawling detectors and superpixel classification framework gains a promising result

on semantic scene segmentation task. Ouyang et al. proposed a deformable deep CNN (DeepID-Net) [87] which introduces a novel deformation constrained pooling (def-pooling) layer to impose geometric penalty on the deformation of various Crawling parts and makes an ensemble of models with different settings. Lenc et al. [88] provided an analysis on the role of proposal generation in CNN-based detectors and tried to replace this stage with a constant and trivial region generation scheme. The goal is achieved by biasing sampling to match the statistics of the ground truth bounding boxes with K-means clustering. However, more candidate boxes are required to achieve comparable results to those of R-CNN.

2) SPP-net: FC layers must take a fixed-size input. That's why R-CNN chooses to warp or crop each region proposal into the same size. However, the Crawling 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 Crawlings vary.

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

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

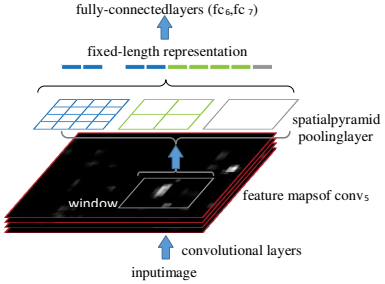


Fig.4.ThearchitectureofSPP-netforCrawlingdetection[64].

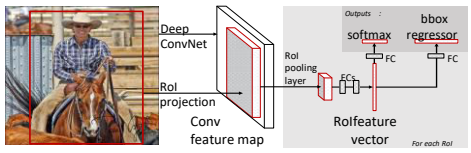


Fig. 5. The architecture of Fast R-CNN [16].

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

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

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 multi-stage pipeline as R-CNN, including feature extraction, network fine-tuning, SVM training and boundingbox regressor fitting. So an additional expense on storage space is still required. Additionally, 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 multi-task 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 Figure 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 a region of interest (RoI) 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 Crawling classes plus one 'background' class) and the other output layer encodes refined boundingbox positions with four real-valued numbers. All parameters in these procedures (except the generation of region proposals) are optimized via a multi-task loss in an end-to-end way.

The multi-tasks loss L is defined as below to jointly train classification and bounding-box regression,

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

where $L_{cls}(p,u) = -\log p_u$ 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. $L_{loc}(t^u,v)$ 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,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 Crawling proposal with a log-space height/width shift and scaleinvariant 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 below

$$L_{loc}(t^u, v) = \sum_{i \in x,y,w,h} smooth_{L_1}(t_i^u - v_i) \quad (2)$$

where

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

To accelerate the pipeline of Fast R-CNN, another two tricks are of necessity. On one hand, if training samples (i.e. RoIs) come from different images, back-propagation through the SPP layer becomes highly inefficient. Fast R-CNN samples mini-batches 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 multi-task loss. It saves the additional 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 Crawling 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. introduced an additional Region Proposal Network (RPN) [18], [92], which acts in a nearly cost-free way by sharing full-image conv features with detection network.

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

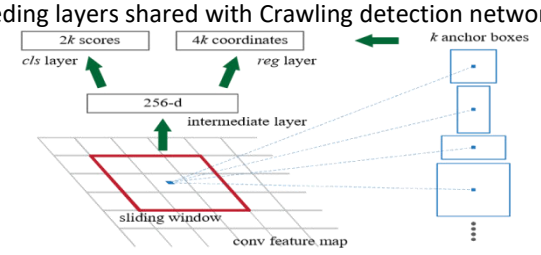


Fig. 6. The RPN in Faster R-CNN [18]. K predefined anchor boxes are convoluted with each sliding window to produce fixed-length vectors which are taken by cls and reg layer to obtain corresponding outputs.

The architecture of RPN is shown in Figure 6. The network slides over the conv feature map and fully connects to an $n \times n$ spatial window. A low dimensional vector (512-d 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 non-linearity, ReLU is applied to the output of theThe regressions towards true bounding boxes are achieved$n \times n$ conv layer.

by comparing proposals relative to reference boxes (anchors). In the Faster R-CNN, anchors of 3 scales and 3 aspect ratios are adopted. The loss function is similar to (1).

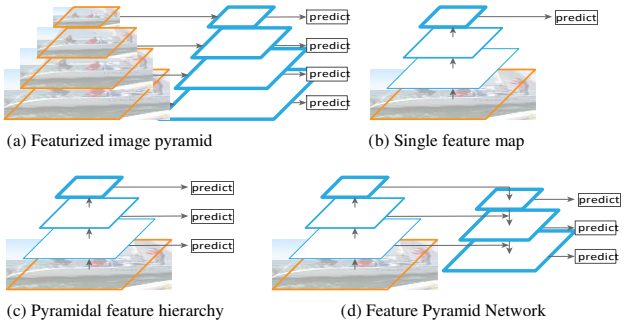
$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (4)$$

where p_i shows the predicted probability of the i-th anchor being an Crawling. The ground truth label p_{*i} is 1 if the anchor is positive, otherwise 0. t_i stores 4 parameterized coordinates of the predicted bounding box while t_{*i} is related to the groundtruth 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 mini-batch size (N_{cls}) and the number of anchor locations (N_{reg}), respectively. In the form of fully-convolutional networks, Faster R-CNN can be trained end-to-end by back-propagation and SGD in an alternate training manner.

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

5) R-FCN: Divided by the RoI pooling layer, a prevalent family [16], [18] of deep networks for Crawling detection are 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 Residual Nets (ResNets) [47] and GoogLeNets [45], [93], are fully convolutional. To adapt to these architectures, it's



(a) Featurized image pyramid (b) Single feature map

(c) Pyramidal feature hierarchy (d) Feature Pyramid Network

Fig. 7. The main concern of FPN [66]. (a) It is slow to use an image pyramid to build a feature pyramid. (b) Only single scale features is adopted for faster detection. (c) An 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.

natural to construct a fully convolutional Crawling detection network without RoI-wise subnetwork. However, it turns out to be inferior with such a naive solution [47]. This inconsistence is due to the dilemma of respecting translation variance in Crawling detection compared with increasing translation invariance in image classification. In other words, shifting an Crawling inside an image should be indiscriminative in image classification while any translation of an Crawling in a bounding box may be meaningful in Crawling detection. A manual insertion of the RoI pooling layer into convolutions can break down translation invariance at the expense of additional unshared region-wise layers. So Li et al. [65] proposed a region-based fully convolutional networks (R-FCN, Fig. S2).

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 $k \times k$ firstly 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 bounding boxes.

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

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

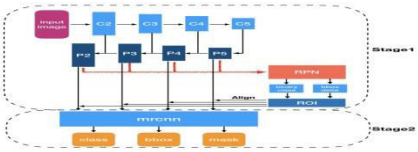


Fig. 8. The Mask R-CNN framework for instance segmentation [67].

resolution maps of the feature hierarchy.

Different from these approaches, FPN [66] holds an architecture with a bottom-up pathway, a top-down pathway and several lateral connections to combine low-resolution and semantically strong features with high-resolution and semantically weak features (Figure 7(d)). The bottom-up 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 top-down pathway.

To build the top-down pathway, feature maps from higher network stages are upsampled at first and then enhanced with those of the same spatial size from the bottom-up 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 element-wise 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, state-of-theart 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 Crawling 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 Crawlings in an image and segmenting each instance (semantic segmentation [97]). These two tasks are usually regarded as two independent processes. And the multi-task 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

Volume: 04 Issue: 07 | July -2020 ISSN: 2582-3930

bounding box regression, the Mask R-CNN [67] adds a branch to predict segmentation masks in a pixel-to-pixel manner (Figure 8).

Different from the other two branches which are inevitably collapsed into short output vectors by FC layers, the segmentation mask branch encodes an $m \times m$ mask to maintain the explicit Crawling spatial layout. This kind of fully convolutional representation requires fewer parameters but is more accurate than that of [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 multi-task loss. An 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 pixelto-pixel mask prediction. To solve this problem, Mask R-CNN adopts an easy and quantization-free layer, namely RoIAlign, to preserve the specific per-pixel spatial correspondence faithfully. RoIAlign is achieved by replacing the tough quantization of RoI pooling with bilinear interpolation [99], computing the precise 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 alittle computational burden and its cooperation with other tasks provides complementary information for Crawling detection. As a result, Mask R-CNN is straightforward to implement with promising instance segmentation and Crawling detection results. In a word, Mask R-CNN may be a flexible and efficient framework for instance-level recognition, which may be easily generalized to other tasks (e.g. human pose estimation [7][S4]) with minimal modification.

8) Multi-task Learning, Multi-scale Representation and Contextual Modelling: Although the Faster R-CNN gets promising results with several hundred proposals, it still struggles in small-size Crawling detection and localization, mainly thanks to the coarseness of its feature maps and limited information provided especially candidate boxes. The phenomenon is more obvious on the Microsoft COCO dataset which consists of Crawlings at a broad range of scales, less prototypical images, and requires more precise localization. To tackle these problems, it's necessarily to accomplish Crawling detection with multi-task learning [100], multi-scale representation [95] and context modelling [101] to mix complementary information from multiple sources.

Multi-task Learning learns a useful representation for multiple correlated tasks from an equivalent input [102], [103]. Brahmbhatt et al. introduced conv features trained for Crawling segmentation and 'stuff' (amorphous categories like ground and water) to guide accurate Crawling detection of small Crawlings (StuffNet) [100]. 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. incorporated the weakly-supervised Crawling segmentation cues and region-based Crawling detection into a multi-stage architecture to completely exploit the learned segmentation features [104].

Multi-scale Representation combines activations from multiple layers with skip-layer connections to supply semantic information of various spatial resolutions [66]. Cai et al. proposed the MS-CNN [105] to ease the inconsistency between the sizes of Crawlings and receptive fields with multiple scale-independent output layers. Yang et al. investigated two strategies, namely scale-dependent pooling (SDP) and layerwise cascaded rejection classifiers (CRC), to take advantage of appropriate scale-dependent conv features [33]. Kong et al. proposed the HyperNet to calculate the shared features between RPN and Crawling detection network by aggregating and compressing hierarchical feature maps from different resolutions into a consistent space [101].

Contextual Modelling improves detection performance by exploiting features from or around RoIs of various support regions and resolutions to affect occlusions and native similarities [95]. Zhu et al. proposed the SegDeepM to take advantage of Crawling segmentation which reduces the dependency on initial candidate boxes with Markov Random Field [106]. Moysset et al. took advantage of 4 directional 2D-LSTMs [107] to convey global context between different local regions and reduced trainable parameters with local parameter-sharing [108]. Zeng et al. proposed a completely unique GBD-Net by introducing gated functions to regulate message transmission between different support regions [109].

The Combination incorporates different components above into an equivalent model to enhance detection performance further. Gidaris et al. proposed the Multi-Region CNN (MR-CNN) model [110] to capture different aspects of an Crawling, the distinct appearances of varied Crawling parts and semantic segmentation-aware features. to get contextual and multiscale representations, Bell et al. proposed the Inside-Outside Net (ION) by exploiting information both inside and out of doors the RoI [95] with spatial recurrent neural networks [111] and skip pooling [101]. Zagoruyko et al. proposed the MultiPath architecture by introducing three modifications to the Fast R-CNN [112], including multi-scale skip connections [95], a modified foveal structure [110] and a

completely unique loss function summing different IoU losses.

9) Thinking in Deep Learning based Crawling Detection: aside from the above approaches, there are still many important factors for continued progress.

There is an outsized imbalance between the amount of annotated Crawlings and background examples. to deal with this problem, Shrivastava et al. proposed an efficient online mining algorithm (OHEM) [113] for automatic selection of the hard examples, which results in a simpler and efficient training.

Instead of concentrating on feature extraction, Ren et al. made an in depth analysis on Crawling classifiers [114], and located that it's of particular importance for Crawling detection to construct a deep and convolutional per-region classifier carefully, especially for ResNets [47] and GoogLeNets [45].

Traditional CNN framework for Crawling detection isn't skilled in handling significant scale variation, occlusion or truncation, especially when only 2D Crawling detection is involved. to deal with this problem, Xiang et al. proposed a completely unique subcategory-aware region proposal network [60], which guides the generation of region proposals with subcategory information associated with Crawling poses and jointly optimize Crawling detection and subcategory classification.

Ouyang et al. found that the samples from different classes follow a longtailed distribution [115], which indicates that different classes with distinct numbers of samples have different degrees of impacts on feature learning. to the present end, Crawlings are firstly clustered into visually similar class groups, then a hierarchical feature learning scheme is adopted to find out deep representations for every group separately.

In order to attenuate 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. proposed the architecture of PVANET [116], which adopts some building blocks including concatenated ReLU [117], Inception [45], and HyperNet [101] to scale back the expense on multi-scale feature extraction and trains the network with batch normalization [43], residual connections [47], and learning rate scheduling supported plateau detection [47]. The PVANET achieves the state-of-the-art performance and may 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 recent end-to-end module Faster R-CNN, an alternate training remains required to get shared convolution parameters between RPN and detection network. As a result, the time spent in handling different components becomes the bottleneck in realtime application.

One-step frameworks supported global regression/classification, mapping straightly from image pixels to bounding box coordinates and sophistication probabilities, can reduce time expense. We firstly reviews some pioneer CNN models, then specialise in two significant frameworks, namely you simply look once (YOLO) [17] and Single Shot MultiBox Detector (SSD) [71].

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

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

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

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

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

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

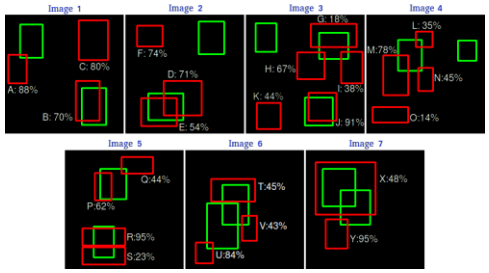


Fig. 9. Main idea of YOLO [17].

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

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

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

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

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

During training, the following loss function is optimized,

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

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

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

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

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

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

Integrating with hard negative mining, data augmentation and a larger number of carefully chosen default anchors, SSD significantly outperforms the Faster R-CNN in terms of accuracy on PASCAL VOC and COCO, while being three times faster. The SSD300 (input image size is 300×300) runs at 59 FPS, which is more accurate and efficient than YOLO. However, SSD is not skilled at dealing with small Crawlings, which can be relieved by adopting better feature extractor backbone (e.g. ResNet101), adding deconvolution layers with

skip connections to introduce additional large-scale context [73] and designing better network structure (e.g. Stem Block and Dense Block) [74].

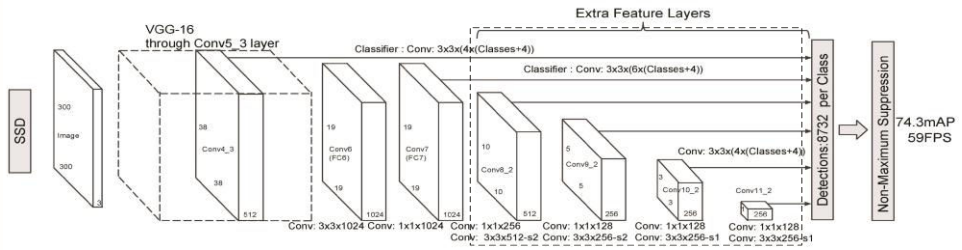


Fig. 10. The architecture of SSD 300 [71]. SSD adds several feature layers to the end of VGG16 backbone network to predict the offsets to default anchor boxes and their associated confidences. Final detection results are obtained by conducting NMS on multi-scale refined bounding boxes.

C. Experimental Evaluation

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

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

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

•end multi-task architecture (FRCN) and RPN (Faster R-

With the introduction of SPP layer (SPP-net), end-to-

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

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

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

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

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

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

different degrees of IoUs and on different Crawling sizes. The results are shown in Table IV.

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

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

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

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

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

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

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

TABLE I
AN OVERVIEW OF PROMINENT GENERIC CRAWLING DETECTION ARCHITECTURES.

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

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

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

Methods	Trained on	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN (Alex) [15]	07	68.1	72.8	56.8	43.0	36.8	66.3	74.2	67.6	34.4	63.5	54.5	61.2	69.1	68.6	58.7	33.4	62.9	51.1	62.5	68.6	58.5
R-CNN(VGG16) [15]	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 [33]	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 [110]	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 [18]	07
70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6	69.9 Faster R-CNN [18]	07+12	76.5
79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6	73.2			
Faster R-CNN [18]	07+12+COCO	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9	78.8 SSD300 [71] 07+12+COCO
80.9	86.3	79.0	76.2	57.6	87.3	88.2	88.6	60.5	85.4	76.7	87.5	89.2	84.5	81.4	55.0	81.9	81.5	85.9	78.9	79.6		
SSD512 [71]	07+12+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 [16]	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 [110]	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 [18] 07++12
84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5	70.4		
Faster R-CNN [18]	07++12+coco	87.4	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 [17]	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 [17]	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	86.5	54.7	83.8	77.3	89.2	76.5	79.2	
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) [16]	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.

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	23.0	43.2	23.0	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
NOC+FRCN(VGG16) [114]	train	21.2	41.5	19.7	-	-	-	-	-	-	-	-	-
NOC+FRCN(Google) [114]	train	24.8	44.4	25.2	-	-	-	-	-	-	-	-	-
NOC+FRCN (ResNet101) [114]	train	27.2	48.4	27.6	-	-	-	-	-	-	-	-	-
GBD-Net [109]	train	27.0	45.8	-	-	-	-	-	-	-	-	-	-
OHEM+FRCN [113]	train		22.6			42.5	22.2	5.0	23.7	34.6		-	
-	-	-		-		-				OHEM+FRCN* [113]			
train	24.4		44.4	24.8	7.1	26.4	37.9		-			-	
-	-	-		-		-			-				
OHEM+FRCN* [113]	trainval	25.5	45.9	26.1	7.4	27.7	38.5	-	-	-	-	-	-
Faster R-CNN [18]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
YOLOv2 [72]	trainval35k	21.6			44.0	19.2	5.0	22.4	35.5	20.7	31.6	33.3	9.8 36.5
54.4 SSD300 [71]	trainval35k	23.2			41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6 37.6
			56.5										
SSD512 [71]	trainval35k	26.8	46.5	27.8	9.0	28.9	41.9	24.8	37.5	39.8	14.0	43.5	59.0
R-FCN (ResNet101) [65]	trainval	29.2	51.5	-	10.8	32.8	45.0	-	-	-	-	-	-
R-FCN*(ResNet101) [65]	trainval	29.9	51.9	-	10.4	32.4	43.3	-	-	-	-	-	-
R-FCN**(ResNet101) [65]	trainval	31.5	53.2	-	14.3	35.5	44.2	-	-	-	-	-	-
Multi-path [112]	trainval	33.2	51.9	36.3	13.6	37.2	47.8	29.9	46.0	48.3	23.4	56.0	66.4
FPN (ResNet101) [66]	trainval35k	36.2	59.1	39.0	18.2	39.0	48.2	-	-	-	-	-	-
Mask (ResNet101+FPN) [67]	trainval35k	38.2	60.3	41.7	20.1	41.1	50.2	-	-	-	-	-	-
Mask (ResNeXt101+FPN) [67]	trainval35k	39.8	62.3	43.4	22.1	43.2	51.2	-	-	-	-	-	-
DSSD513 (ResNet101) [73]	trainval35k	33.2	53.3	35.2	13.0	35.4	51.1	28.9	43.5	46.2	21.8	49.1	66.4
DSOD300 [74]	trainval	29.3	47.3	30.6	9.4	31.5	47.0	27.3	40.7	43.0	16.7	47.1	65.0

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

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* [24]	07	63.1	2.3	0.44
SS+FRCN [16]	07+12	66.9	1.72	0.6
SDP+CRC [33]	07	68.9	0.47	2.1
SS+HyperNet* [101]	07+12	76.3	0.20	5
MR-CNN&S-CNN [110]	07+12	78.2	30	0.03
ION [95]	07+12+S	79.2	1.92	0.5
Faster R-CNN(VGG16) [18]	07+12	73.2	0.11	9.1
Faster R-CNN(ResNet101) [18]	07+12	83.8	2.24	0.4
YOLO [17]	07+12	63.4	0.02	45 SSD300 [71]
07+12	74.3	0.02	46	
SSD512 [71]	07+12	76.8	0.05	19
R-FCN(ResNet101) [65]	07+12+coco	83.6	0.17	5.9
YOLOv2(544*544) [72]	07+12	78.6	0.03	40
DSSD321(ResNet101) [73]	07+12	78.6	0.07	13.6
DSOD300 [74]	07+12+coco	81.7	0.06	17.4
PVANET+ [116]	07+12+coco	83.8	0.05	21.7
PVANET+(compress) [116]	07+12+coco	82.9	0.03	31.3

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

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

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

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

•network (ResNet101 against VGG16) and this time con-It takes more time to train a more complex and deeper sumption can be reduced by adding as many layers into shared fully convolutional layers as possible (FRCN). • Regression based models can usually be processed in realtime at the cost of a drop in accuracy compared with region proposal based models. Also, region proposal based

models can be modified into real-time systems with the introduction of other tricks [116] (PVANET), such as BN [43], residual connections [123].

IV. Salient Crawling Detection

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 pre-computed on the union of generated binary mask B and ground truth Z, F-measure is defined as below

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

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

The MAE score

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

TABLE VI COMPARISON BETWEEN STATE OF THE ART METHODS.

e is computed with the following equation

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 Crawlings over detected non-salient pixels [159].

The following approaches are evaluated: CHM [150], RC [151], DRFI [152], MC [138], MDF [146], LEGS [136], DSR [149], MTDNN [141], CRPSD [142], DCL [143], ELD [153], NLDF [154] and 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 shown 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 recurrent network and semantic segmentation, which provide insights for future improvements. CPRSD, DCL, NLDF and DSSC are all based on multi-scale representations and superpixel segmentation, which provide robust salient regions and smooth boundaries. DCL, NLDF and DSSC perform the best on these four datasets. DSSC earns the best performance by modelling scale-to-scale shortconnections.

Overall, as CNN mainly provides salient information in local regions, most of 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's necessary to strengthen local connections between different CNN layers and as well to utilize complementary information from local and global context.

V. Cr awl Detection

took this histogram to guide the zoom-in and zoomout of the image [171]. Since the crawls are approximately in uniform scale after zoom, compared with other state-ofthe-art baselines, better performance is achieved with less computation cost. Besides, some generic detection frameworks

Dataset	Metrics	CHM [150]	RC [151]	DRFI [152]	MC [138]	MDF [1
				0.222	0.225	0.221	0.147	0.14
ECSSD	wF_β	0.722	0.741	0.787	0.822	0.833	0.827	
	MAE	0.195	0.187	0.166	0.107	0.108	0.118	
HKU-IS	wF_β	0.728	0.726	0.783	0.781	0.860	0.770	
	MAE	0.158	0.165	0.143	0.098	0.129	0.118	
SOD	wF_β	0.655	0.657	0.712	0.708	0.785	0.707	
	MAE	0.249	0.242	0.215	0.184	0.155	0.205	

The bigger $wF\beta$ is or the smaller MAE is, the better the performance is.

are extended to crawl detection with different modifications, e.g. Faster R-CNN [29], [172], [173].

Some authors trained CNNs with other complementary tasks, such as 3D modelling and crawl landmarks, in a multitask learning manner. Huang et al. proposed a unified endto-end FCN framework called DenseBox to jointly conduct crawl detection and landmark localization [174]. Li et al. [175] proposed a multi-task discriminative learning framework which integrates a ConvNet with a fixed 3D mean crawl model in an end-to-end manner. In the framework, two issues are addressed to transfer from generic Crawling detection to crawl detection, namely eliminating predefined anchor boxes by a 3D mean crawl model and replacing RoI pooling layer with a configuration pooling layer. Zhang et al. [176] proposed a deep cascaded multi-task framework named MTCNN which exploits the inherent correlations between crawl detection and alignment in 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 mobile platform, Kalinovskii and Spitsyn proposed a new solution of frontal crawl detection based on compact CNN cascades [177]. This method takes a cascade of three simple CNNs to

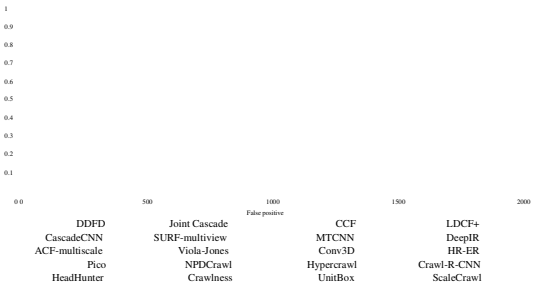
generate, classify and refine candidate Crawling positions progressively. To reduce the effects of large pose variations, Chen et al. proposed a cascaded CNN denoted by Supervised Transformer Network [31]. This network takes a multi-task RPN to predict candidate crawl regions along with associated facial landmarks simultaneously, and adopts a generic R-CNN to verify the existence of valid crawls. Yang et al. proposed a three-stage cascade structure based on FCNs [8], while in each stage, a multi-scale FCN is utilized to refine the positions of possible crawls. Qin et al. proposed a unified framework which achieves better results with the complementary information from different jointly trained CNNs [178].

A. Experimental Evaluation

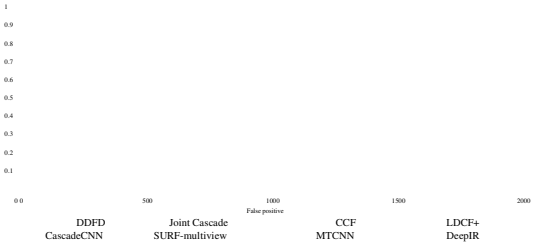
The FDDB [179] dataset has a total of 2,845 pictures in which 5,171 crawls are annotated with elliptical shape. Two types of evaluations are used: the discrete score and continuous score. By varying the threshold of the decision rule, the ROC curve for the discrete scores can reflect the dependence of the detected crawl 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 crawl localization quality.

The evaluated models cover DDFD [168], CascadeCNN [180], ACF-multiscale [181], Pico [182], HeadHunter [183], Joint Cascade [30], SURF-multiview [184], Viola-Jones [166], NPDCrawl [185], Crawlness [169], CCF [186], MTCNN [176], Conv3D [175], Hypercrawl [187], UnitBox [167], LDCF+ [S2], DeepIR [173], HR-ER [188], Crawl-R-CNN [172] and Scale-Crawl [170]. ACF-multiscale, Pico, HeadHunter, Joint Cascade, SURF-multiview, Viola-Jones, NPDCrawl 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 Figure 11.

DSR [149]	MTDNN [141]	CRPSD [142]	DCL [143]	ELD [153]	NLDF [154]	DSSC [155]
0.697	0.818	0.776	0.822	0.767	0.831	
	0.830 0.128	0.170	0.063	0.108	0.121	
	0.099	0.080				
0.872	0.810	0.849	0.898	0.865	0.905	
	0.915 0.037	0.160	0.046	0.071	0.098	
	0.063	0.052				
0.833	-	0.821	0.907	0.844	0.902	0.913
0.040	-	0.043	0.048	0.071	0.048	0.039
-	0.781	-	0.832	0.760	0.810	0.842
-	0.150	-	0.126	0.154	0.143	0.118



(a) Discrete ROC curves



(b) Continuous ROC curves

Fig. 11. The ROC curves of state-of-the-art methods on FDDB.

From Figure 11(a), in spite of relatively competitive results produced by LDCF+, it can be observed that most of classic methods perform with similar results and are outperformed by CNN based methods by a significant margin. From Figure 11(b), it can be observed that most of 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 crawl regions, which is efficient but inaccurate. Crawlness combines the decisions from different part detectors, resulting in precise crawl localizations while being time-consuming. The outstanding performance of MTCNN, Conv3D and Hypercrawl proves the effectiveness of multi-task learning. HR-ER and ScaleCrawl adaptively detect crawls of different scales, and make a balance between accuracy and efficiency. DeepIR and Crawl-R-CNN are two extensions of the Faster R-CNN architecture to crawl 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 multi-task shared CNN features.

VI. CRAWLING OBJECT DETECTION

Recently, crawling object detection has been intensively studied, which has a close relationship to crawling object tracking [189], [190], person re-identification [191], [192] and robot navigation [193], [194]. Prior to the recent progress in DCNN based methods [195], [196], some researchers combined boosted decision forests with hand-crafted features to obtain crawling object 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 crawling object instances of small sizes in typical scenarios of crawling object detection (e.g. automatic driving and intelligent surveillance), the application of RoI pooling layer in generic Crawling detection pipeline may result in 'plain' features due to collapsing bins. In the meantime, the main source of false predictions in crawling object detection is the confusion of hard background instances, which is in contrast to the interference from multiple categories in generic Crawling detection. As a result, different configurations and components are required to accomplish accurate crawling object detection.

A. Deep learning in Crawling object Detection

Although DCNNs have obtained excellent performance on generic Crawling 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 researches have been conducted to analyze the reasons. Zhang et al. attempted to adapt generic Faster R-CNN [18] to crawling object detection [203]. They modified the downstream classifier by adding boosted forests to shared, highresolution conv feature maps and taking a RPN to handle small instances and hard negative examples. To deal with complex occlusions existing in crawling object images, inspired by DPM [24], Tian et al. proposed a deep learning framework called DeepParts [204], which makes decisions based 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. proposed multi-spectral deep neural networks for crawling object detection to combine complementary information from color and thermal images [205]. Tian et al. [206] proposed a taskassistant CNN (TA-CNN) to jointly learn multiple tasks with multiple data sources and to combine crawling object attributes with semantic scene attributes together. Du et al. proposed a deep neural network fusion architecture for fast and robust crawling object detection [207]. Based on the candidate bounding boxes generated with SSD detectors [71], multiple binary classifiers are processed parallelly to conduct soft-rejection based network fusion (SNF) 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]. So the modification and simplification is of significance to reduce the burden on both software and hardware to satisfy real-time detection demand. Tome et al. proposed a novel solution to adapt generic Crawling detection pipeline to crawling object detection by optimizing most of its stages [59]. 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 labelling 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. addressed the problem of Human-Aware Navigation [32] and proposed a vision-based person tracking system guided by multiple camera sensors.

TABLE VII
DETAILED BREAKDOWN PERFORMANCE COMPARISONS OF STATE-OF-THE-ART MODELS ON CALTECH CRAWLING OBJECT DATASET. 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

B. Experimental Evaluation

The evaluation is conducted on the most popular Caltech Crawling object dataset [3]. The dataset was collected from the videos of a vehicle driving through an urban environment and consists of 250,000 frames with about 2300 unique crawling objects and 350,000 annotated bounding boxes (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 (FPPI) [3]. According to the differences in the height and visible part of the BBs, a total of 9 popular settings are adopted to evaluate different properties of these models. Details of these settings are as [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 crawling object 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 multi-scale representation of MS-CNN improves accuracy of crawling object locations. SA-FastRCNN extends Fast R-CNN to automatically detecting crawling objects 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 crawling objects. 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 multi-level semantic information for identifying and locating crawling objects. 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 multi-scale 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 Crawling detection, there are still many open issues for future work.

The first one is small Crawling detection such as occurring in COCO dataset and in crawl detection task. To improve localization accuracy on small Crawlings under partial occlusions, it is necessary to modify network architectures from the following aspects.

- *mation fusion.Multi-task joint optimization and multi-modal infor*-Due to the correlations between different tasks within and outside Crawling detection, multi-task joint optimization has already been studied by many researchers [16] [18]. However, apart from the tasks mentioned in Subs. III-A8, it is desirable to think over the characteristics of different sub-tasks of Crawling detection (e.g. superpixel semantic segmentation in salient Crawling detection) and extend multi-task optimization to other applications such as instance segmentation [66], multi-Crawling tracking [202] and multi-person pose estimation [S4]. Besides, given 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.

- which is more apparent in crawl detection and crawling object*Scale adaption.* Crawlings usually exist in different scales, detection. To increase the robustness to scale changes, it is demanded to train scale-invariant, multi-scale or scaleadaptive detectors. For scale-invariant detectors, more powerful backbone architectures (e.g. ResNext [123]), negative sample mining [113], reverse connection [213] and subcategory modelling [60] are all beneficial. For multi-scale detectors, both the FPN [66] which produces multi-scale feature maps and Generative Adversarial Network [214] which narrows representation differences between small Crawlings 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 Crawlings adaptively.

- distribution plays an important role in Crawling detection. So*Spatial correlations and contextual modelling.* Spatial region proposal generation and grid regression are taken to obtain probable Crawling locations. However, the correlations between multiple proposals and Crawling categories are ignored. Besides, 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 Crawling detection, with the emergence of large-scale image and video data. The following three aspects can be taken into account.

•detectors are built in different stages or layers [180], [220].*Cascade network.* In a cascade network, a cascade of

And easily distinguishable examples are rejected at shallow layers so that features and classifiers at latter 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. So the optimizations of different CNNs are isolated, which stresses the necessity of end-toend optimization for CNN cascade. At the same time, it is also a matter of concern to build contextual associated cascade networks with existing layers. • *Unsupervised and weakly supervised learning.* It's very time consuming to manually draw large quantities of bounding boxes. To release this burden, semantic prior [55], unsupervised Crawling discovery [221], multiple instance learning [222] and deep neural network prediction [47] can be integrated to make best use of image-level supervision to assign Crawling category tags to corresponding Crawling regions and refine Crawling 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.

•platforms, it is significant to make a balance among speed,*Network optimization.* Given specific applications and

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 fewer number of parameters [209]. And this situation can be relieved by introducing better pre-training 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 2D Crawling detection to adapt 3D Crawling detection and video Crawling detection, with the requirements from autonomous driving, intelligent transportation and intelligent surveillance.

• *3D Crawling detection*. With the applications of 3D sensors (e.g. LIDAR and camera), additional depth information can be utilized to better understand the images in 2D and extend the image-level knowledge to the real world. However, seldom of these 3D-aware techniques aim to place correct 3D bounding boxes around detected Crawlings. To achieve better bounding results, multi-view representation [181] and 3D proposal network [228] may provide some guidelines to encode depth information with the aid of inertial sensors (accelerometer and gyrometer) [229].

•different frames play an important role in understanding*Video Crawling detection.* Temporal information across the behaviors of different Crawlings. However, the accuracy suffers from degenerated Crawling 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 Crawling 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 Crawling detection has been a research hotspot in recent years. This paper provides a detailed review on deep learning based Crawling detection frameworks which handle different sub-problems, such as occlusion, clutter and low resolution, with different degrees of modifications on R-CNN. The review starts on generic Crawling detection pipelines which provide base architectures for other related tasks. Then, three other common tasks, namely salient Crawling detection, crawl detection and crawling object detection, are also briefly reviewed. Finally, we propose several promising future directions to gain a thorough understanding of the Crawling 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.

Acknowledgments

This research was supported in under guidance of Mr. Anurag Singh who is an outstanding teacher in Galgotias University. Without his help this project doesn't become possible.

References

[1] P. F. Felzenszwalb, R. B. Girshick, D. Mcallester, and D. Ramanan, "Crawling detection with discriminatively trained part-based

models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, p. 1627, 2010.

[2] K. K. Sung and T. Poggio, "Example-based learning for view-based human crawl detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 1, pp. 39–51, 2002.

[3] C. Wojek, P. Dollar, B. Schiele, and P. Perona, "Crawling object detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 34, no. 4, p. 743, 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, 1996.

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM MM*, 2014.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[7] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.

[8] Z. Yang and R. Nevatia, "A multi-scale cascade fully convolutional network crawl detector," in *ICPR*, 2016.

[9] C. Chen, A. Seff, A. L. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *ICCV*, 2015.

[10] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d Crawling detection network for autonomous driving," in *CVPR*, 2017.

[11] A. Dundar, J. Jin, B. Martini, and E. Culurcello, "Embedded streaming deep neural networks accelerator with applications," *IEEE Trans. Neural Netw. & Learning Syst.*, vol. 28, no. 7, pp. 1572–1583, 2017.

[12] R. J. Cintra, S. Duffner, C. Garcia, and A. Leite, "Low-complexity approximate convolutional neural networks," *IEEE Trans. Neural Netw. & Learning Syst.*, vol. PP, no. 99, pp. 1–12, 2018.

[13] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data." *IEEE Trans. Neural Netw. & Learning Syst.*, vol. PP, no. 99, pp. 1–15, 2017.

[14] A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature extraction with deep neural networks by a generalized discriminant analysis." *IEEE Trans. Neural Netw. & Learning Syst.*, vol. 23, no. 4, pp. 596–608, 2012.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate Crawling detection and semantic segmentation," in *CVPR*, 2014.

[16] R. Girshick, "Fast r-cnn," in *ICCV*, 2015.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time Crawling detection," in *CVPR*, 2016.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards realtime Crawling detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.

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

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

[21] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid Crawling detection," in *ICIP*, 2002.

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

[23] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," *J. of Comput. & Sys. Sci.*, vol. 13, no. 5, pp. 663–671, 1997.

[24] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Crawling detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, pp. 1627–1645, 2010.

[25] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual Crawling classes challenge 2007 (voc 2007) results (2007)," 2008.

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[27] N. Liu, J. Han, D. Zhang, S. Wen, and T. Liu, "Predicting eye fixations using convolutional neural networks," in *CVPR*, 2015.

[28] E. Vig, M. Dorr, and D. Cox, "Large-scale optimization of hierarchical features for saliency prediction in natural images," in *CVPR*, 2014.

[29] H. Jiang and E. Learned-Miller, "Crawl detection with the faster r-cnn," in *FG*, 2017.

[30] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun, "Joint cascade crawl detection and alignment," in *ECCV*, 2014.

[31] D. Chen, G. Hua, F. Wen, and J. Sun, "Supervised transformer network for efficient crawl detection," in *ECCV*, 2016.

[32] D. Ribeiro, A. Mateus, J. C. Nascimento, and P. Miraldo, "A real-time crawling object detector using deep learning for human-aware navigation," *arXiv:1607.04441*, 2016.

[33] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate cnn Crawling detector with scale dependent pooling and cascaded rejection classifiers," in *CVPR*, 2016.

[34] P. Druzhkov and V. Kustikova, "A survey of deep learning methods and software tools for image classification and Crawling detection," *Pattern Recognition and Image Anal.*, vol. 26, no. 1, p. 9, 2016.

[35] W. Pitts and W. S. McCulloch, "How we know universals the perception of auditory and visual forms," *The Bulletin of Mathematical Biophysics*, vol. 9, no. 3, pp. 127–147, 1947.

[36] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representation by back-propagation of errors," *Nature*, vol. 323, no. 323, pp. 533–536, 1986.

[37] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Sci.*, vol. 313, pp. 504–507, 2006.

[38] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic moeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, 2012.

[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[40] L. Deng, M. L. Seltzer, D. Yu, A. Acero, A.-r. Mohamed, and G. Hinton, "Binary coding of speech spectrograms using a deep autoencoder," in *INTERSPEECH*, 2010.

[41] G. Dahl, A.-r. Mohamed, G. E. Hinton *et al.*, "Phone recognition with the mean-covariance restricted boltzmann machine," in *NIPS*, 2010.

[42] G. E. Hinton, N. Srivastav, A. Krizhevky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing coadaptation of feature detectors," *arXiv:1207.0580*, 2012.

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

[44] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv:1312.6229*, 2013.

[45] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.

[47] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[48] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[49] M. Oquab, L. Bottou, I. Laptev, J. Sivic *et al.*, "Weakly supervised Crawling recognition with convolutional neural networks," in *NIPS*, 2014.

[50] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014.

[51] F. M. Wadley, "Probit analysis: a statistical treatment of the sigmoid response curve," *Annals of the Entomological Soc. of America*, vol. 67, no. 4, pp. 549–553, 1947.

[52] K. Kavukcuoglu, R. Fergus, Y. LeCun *et al.*, "Learning invariant features through topographic filter maps," in *CVPR*, 2009.

[53] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun, "Learning convolutional feature hierarchies for visual recognition," in *NIPS*, 2010.

[54] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *CVPR*, 2010.

[55] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *ICCV*, 2015.

[56] Z.-Q. Zhao, B.-J. Xie, Y.-m. Cheung, and X. Wu, "Plant leaf identification via a growing convolution neural network with progressive sample learning," in *ACCV*, 2014.

[57] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *ECCV*, 2014.

[58] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, "Deep learning for content-based image retrieval: A comprehensive study," in *ACM MM*, 2014.

[59] D. Tome, F. Monti, L. Baroffio, L. Bondi, M. Tagliasacchi, and` S. Tubaro, "Deep convolutional neural networks for crawling object detection," *Signal Process.: Image Commun.*, vol. 47, pp. 482–489, 2016.

[60] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for Crawling proposals and detection," in *WACV*, 2017.

[61] Z.-Q. Zhao, H. Bian, D. Hu, W. Cheng, and H. Glotin, "Crawling object detection based on fast r-cnn and batch normalization," in *ICIC*, 2017. [62] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *ICML*, 2011.

[63] Z. Wu, X. Wang, Y.-G. Jiang, H. Ye, and X. Xue, "Modeling spatialtemporal clues in a hybrid deep learning framework for video classification," in *ACM MM*, 2015.

[64] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, 2015.

[65] Y. Li, K. He, J. Sun *et al.*, "R-fcn: Crawling detection via region-based fully convolutional networks," in *NIPS*, 2016, pp. 379–387.

[66] T.-Y. Lin, P. Dollar, R. B. Girshick, K. He, B. Hariharan, and S. J.´ Belongie, "Feature pyramid networks for Crawling detection," in *CVPR*, 2017.

[67] K. He, G. Gkioxari, P. Dollar, and R. B. Girshick, "Mask r-cnn," in´ *ICCV*, 2017.

[68] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable Crawling detection using deep neural networks," in *CVPR*, 2014.

[69] D. Yoo, S. Park, J.-Y. Lee, A. S. Paek, and I. So Kweon, "Attentionnet: Aggregating weak directions for accurate Crawling detection," in *CVPR*, 2015.

[70] M. Najibi, M. Rastegari, and L. S. Davis, "G-cnn: an iterative grid based Crawling detector," in *CVPR*, 2016.

[71] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *ECCV*, 2016.

[72] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," *arXiv:1612.08242*, 2016.

[73] C. Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv:1701.06659*, 2017.

[74] Z. Shen, Z. Liu, J. Li, Y. G. Jiang, Y. Chen, and X. Xue, "Dsod: Learning deeply supervised Crawling detectors from scratch," in *ICCV*, 2017.

[75] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming autoencoders," in *ICANN*, 2011.

[76] G. W. Taylor, I. Spiro, C. Bregler, and R. Fergus, "Learning invariance through imitation," in *CVPR*, 2011.

[77] X. Ren and D. Ramanan, "Histograms of sparse codes for Crawling detection," in *CVPR*, 2013.

[78] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for Crawling recognition," *Int. J. of Comput. Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[79] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun, "Crawling object detection with unsupervised multi-stage feature learning," in *CVPR*, 2013.

[80] P. Krahenb¨ uhl and V. Koltun, "Geodesic Crawling proposals," in¨ *ECCV*, 2014.

[81] P. Arbelaez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik,´ "Multiscale combinatorial grouping," in *CVPR*, 2014.

[82] C. L. Zitnick and P. Dollar, "Edge boxes: Locating Crawling proposals´ from edges," in *ECCV*, 2014.

[83] W. Kuo, B. Hariharan, and J. Malik, "Deepbox: Learning Crawlingness with convolutional networks," in *ICCV*, 2015.

[84] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollar, "Learning to´ refine Crawling segments," in *ECCV*, 2016.

[85] Y. Zhang, K. Sohn, R. Villegas, G. Pan, and H. Lee, "Improving Crawling detection with deep convolutional networks via bayesian optimization and structured prediction," in *CVPR*, 2015.