Pattern Recognition 86 (2019) 99–111

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Multi-label learning of part detectors for occluded pedestrian detection

Chunluan Zhou [a,*], Junsong Yuan [b]

[a] Nanyang Technological University, Singapore
[b] State University of New York, USA

ARTICLE INFO

Article history:
Received 29 March 2018
Revised 30 June 2018
Accepted 27 August 2018
Available online 5 September 2018

MSC:
00-01
99-00

Keywords:
Pedestrian detection
Part detectors
Multi-label learning
Occlusion handling
Detector integration
Context

ABSTRACT

Despite recent progress of pedestrian detection, it remains a challenging problem to detect pedestrians that are partially occluded due to the uncertainty and diversity of partial occlusion patterns. Following a commonly used framework of handling partial occlusions by part detection, we propose a multi-label learning approach to jointly learn part detectors to capture partial occlusion patterns. The part detectors share a set of decision trees which are learned and combined via boosting to exploit part correlations and also reduce the computational cost of applying these part detectors for pedestrian detection. The learned decision trees capture the overall distribution of all the parts. When used as a pedestrian detector individually, our part detectors learned jointly show better performance than their counterparts learned separately in different occlusion situations. For occlusion handling, several methods are explored to integrate the part detectors learned by the proposed approach. Context is also exploited to further improve the performance. The proposed approach is applied to hand-crafted channel features and features learned by a deep convolutional neural network, respectively. Experiments on the Caltech and CUHK datasets show state-of-the-art performance of our approach for detecting occluded pedestrians, especially heavily occluded ones.

1. Introduction

Occlusions occur frequently in practical applications, which presents a great challenge for pedestrian detection. Most state-of-the-art pedestrian detection approaches [1–5] train a full-body detector to detect pedestrians which are non-occluded or slightly occluded but ignore situations in which pedestrians are heavily occluded. Although these approaches show promising performance for detecting pedestrians which are not heavily occluded, they often fail to work well when heavy occlusions are present. For example, RPN+BF [3] achieves a log-average miss rate of 9.6% on the *Reasonable* subset of the Caltech dataset [6], but its performance drops dramatically to 74% on the *Heavy* subset in which pedestrian examples are heavily occluded. Since most of the body of a heavily occluded pedestrian is invisible, a full-body detector would probably be misled by the background region inside its detection window so that it tends to miss the pedestrian. As illustrated in Fig. 1(a-b), the heavily occluded pedestrian (Blue bounding box) is only ranked at 5th among the top five detections obtained by a full-body detector. In this paper, we explore how to handle occlusions properly to improve the performance for heavily occluded pedestrian detection.

A common solution to occlusion handling for pedestrian detection is to learn a set of part/occlusion-specific detectors which are then properly integrated for detecting both non-occluded and partially occluded pedestrians [7–14]. This solution is based on the assumption that when a full-body detector fails to detect a partially occluded pedestrian, the detectors of the parts which are not occluded can still have high detection scores on the pedestrian (See Fig. 1(b-c)). For this solution, the reliability of part detectors is of great importance, since part detectors are its building blocks. Usually, part detectors are learned separately [7–11,13,14]. Learning part detectors separately has two drawbacks: (1) Correlations among parts are ignored, which would affect the reliability of the learned part detectors; (2) The computational cost of applying the learned part detectors increases linearly with the number of parts. In [12], part detector learning and integration are achieved jointly using a single convolutional neural network. However, this approach only uses class labels and part detectors are learned implicitly in a weakly supervised fashion. We believe part-level supervision can be exploited to further improve its performance.

To address the above issues of learning part detectors, we propose a multi-label learning approach to jointly learn part detec-

* Corresponding author.
E-mail address: czhou002@e.ntu.edu.sg (C. Zhou).

https://doi.org/10.1016/j.patcog.2018.08.018
0031-3203/

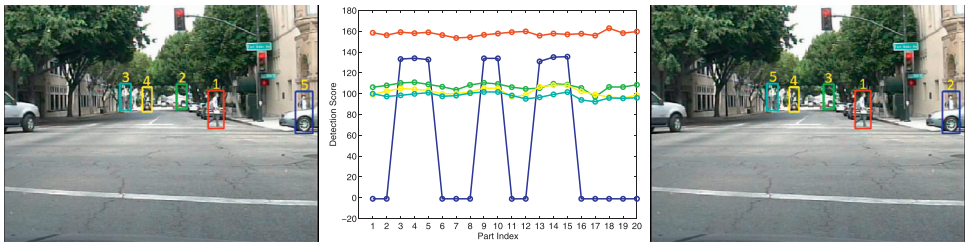


Fig. 1. Occlusion handling. (a) Top five detections of a full-body detector. The heavily occluded pedestrian (Blue bounding box) is only ranked at 5th. (b) Scores of the five detections in (a) given by 20 part detectors. Each curve shows the 20 scores of one detection in the same color. The first detector is the full-body detector. Fig. 2 shows the 20 parts. (c) The five detections in (a) are re-ranked by properly integrating the 20 part detectors. The heavily occluded pedestrian (Blue bounding box) is ranked at 2nd. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

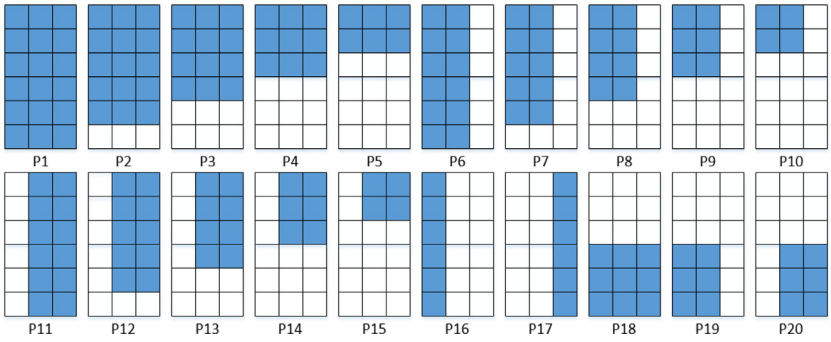


Fig. 2. Part pool. Blue regions denote parts. The first part is the whole body which is modeled as a template of 6×3 cells. Parts 2–17 are designed to handle situations where occlusions occur from the left, right or bottom and the last three parts are used for detecting the lower body. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

tors with part-level supervision. The goal of joint learning is to (1) improve the part detectors by exploiting correlations among parts, e.g. some parts of the body tend to appear/disappear together, and (2) reduce the computational cost of applying the learned part detectors for pedestrian detection. Learning and combining a set of decision trees via boosting to form a boosted forest is widely adopted for pedestrian detection and has shown to work well in practice [2,3,15,16]. We also choose decision trees to form our part detectors. However, instead of learning a set of decision trees for each part detector, we only construct one set of decision trees which are shared by all the part detectors. To exploit part correlations, these decision trees are learned and combined to capture the overall distribution of all the parts. We adapt AdaBoost.MH [17], which is a multi-class, multi-label version of AdaBoost, to learn these decision trees. For occlusion handling, we explore several methods to integrate the part detectors jointly learned by the proposed approach. We also exploit context to further improve the performance. The effectiveness of our approach is demonstrated on the Caltech [6] and CUHK [9] datasets. We apply the proposed multi-label learning approach to channel features [18] and features learned by a convolutional neural network (CNN features) respectively. When used for pedestrian detection individually, the part detectors learned jointly by the proposed approach show better performance than their counterparts learned separately. Using channel features our approach improves state-of-the-arts for detecting pedestrians in different occlusion situations while using CNN features our approach shows comparable performance for detecting pedestrians that are non-occluded or slightly occluded and achieves the best performance for detecting occluded pedestrians, especially heavily occluded ones.

In summary, our contributions are two-fold: (1) we propose a multi-label learning approach to jointly learn part detectors which share decision trees to exploit correlations among parts and also reduce the computational cost of applying these part detectors; (2) we explore several integration methods to integrate the part detectors learned by the proposed approach for occlusion handling and exploit context to further improve detection performance. A short version of this work appears in [19]. The remainder of this paper is organized as follows. Related work is discussed in Section 2. Section 3 presents the proposed multi-label part detector learning approach. How to exploit the learned part detectors for occlusion handling is described in Section 4. Experimental results are given in Section 5. Section 6 concludes the paper.

2. Related work

In the past decade, many efforts have been made to improve pedestrian detection [6,20–26]. Most recent approaches use either hand-crafted channel features or features learned by convolutional neural networks (CNN features). For the approaches using hand-crafted channel features [1,2,15,18,27–29], boosting is usually applied to learn and combine a set of decision trees to form a pedestrian detector and pedestrian detection is carried out in a sliding-window fashion. In [27], multiple registered image channels are computed by applying a set of linear and non-linear transformations, e.g. gradients, Gabor transformations and thresholding, to an input image. Features are extracted by computing sums over local rectangular regions, which can be implemented efficiently via integral images. Then, boosting is applied to select a subset of features for learning a pedestrian detector. Due to promising performance achieved by the combination of channel features and boosting, some approaches are proposed to enrich channel features [1,2,18,29] and accelerate feature extraction [15]. A thorough study of how to build an accurate pedestrian detector using the combination of channel features and boosting is presented in [28]. For the approaches using CNN features [3–5,9,12,14,30–34], a deep convo-

lutional neural network is trained to either form a pedestrian classifier [5,9,12,30,32–34] or generate features which are combined with other types of classifiers for pedestrian detection [3,4,31]. Generally, these approaches perform detection by classifying a set of pedestrian proposals. In [9], a Restricted Boltzmann Machine (RBM) is proposed to integrate detection scores from part detectors to handle occlusions. A deep network architecture is proposed in [12] to integrate feature extraction, deformation handling, occlusion handling and classification in a single network. How to learn a deep convolutional neural network for pedestrian detection is thoroughly studied in [30]. In [33], a deep convolutional neural network is proposed for multi-scale pedestrian detection. Different types of information, e.g. attributes [32], edges and segmentation channels [34], and thermal data [5] are exploited to improve pedestrian detection performance. To validate the effectiveness of our multi-label learning approach, we apply it to both channel and CNN features.

Since occlusions occur frequently in practical applications, many approaches have been proposed to handle occlusions for pedestrian detection. The approach in [35] adopts an implicit shape model to generate a set of pedestrian hypotheses which are further refined using local and global cues to obtain their visible regions. In [36], a pedestrian template is divided into a set of blocks and occlusion reasoning is conducted by estimating the visibility status of each block. A probabilistic framework [37] is proposed to exploit multi-pedestrian detectors to aid single-pedestrian detectors, which can handle partial occlusions especially when pedestrians occlude each other. In [38,39], a set of occlusion patterns are discovered to capture occlusions of single pedestrians and multiple pedestrians and then a deformable part model [40,41] is employed to learn a mixture of occlusion-specific detectors. A widely used occlusion handling strategy for pedestrian detection is to learn a set of part detectors which are then fused properly for detecting partially occluded pedestrians [7–14,42,43]. Most of these approaches [7–11,13,14] focus on how to integrate part detectors. Part detectors are integrated by linear combination [7,13,14], rules [8,10] and Restricted Boltzmann Machines [9,11]. Our approach also adopts a similar framework. Different from these approaches in which part detectors are usually learned separately, part detectors are learned jointly in our approach so as to exploit part correlations for improving these detectors and reduce the computational cost of applying the learned part detectors for pedestrian detection.

Contextual information is commonly used to improve the performance for general object detection [40,44–48]. There are also several specific approaches which exploit context for pedestrian detection [49–52]. In [52], pedestrian-vehicle relationships are exploited to suppress false positives located around vehicles in traffic scenes. Cascaded classifiers are learned by boosting [49] or deep learning [50] in a sequential way that the outputs of a classifier provide context for the classifier at the next stage. In [51], the output from a two-pedestrian detector is taken as context for a single-pedestrian detector to improve the performance of the latter using a probabilistic framework. Different from these approaches, we exploit context by analyzing the detection distribution of a pedestrian detector.

3. Multi-label learning of part detectors

3.1. Part representation

We model the whole body of a pedestrian as a rectangular template without distinguishing different viewpoints. The template is divided into an $H \times W$ grid. Any rectangular sub-region in the template is considered as a part. Mathematically, a part can be expressed as a 4-tuple $p = (l, t, r, b)$, where (l, t) and (r, b) are the coordinates of the top-left and bottom-right corners of the part respectively with $0 \le l < r \le W$ and $0 \le t < b \le H$. In our implementation, we set $H = 6$ and $W = 3$. According to prior knowledge that pedestrians are usually occluded from the left, right or bottom, we manually design a pool of parts as shown in Fig. 2. The part pool can be expressed as $\mathcal{P} = \{p_k | 1 \le k \le K\}$, where $p_k = (l_k, t_k, r_k, b_k)$ and $K = 20$. For pedestrian detection, images are usually represented by a set of feature maps, e.g. locally decorrelated channel features (LDCF) [18] and convolutional channel features (CCF) [31]. To represent a part on a pedestrian, a direct way is to crop regions that correspond to the part from the feature maps. One problem of this representation is that small parts on a pedestrian are difficult to be reliably detected as the information from the small parts is relatively limited compared with that from large parts, especially when the pedestrian is small. To address this problem, we represent a part using features from the whole body instead. Features from the surrounding region of the part can be taken as its context. In this way, all the parts have the same representation.

3.2. Multi-label formulation

Let $\mathcal{X}$ be an instance space which consists of image regions. For each part $p_k \in \mathcal{P}$, we want to learn a detector $d_k : \mathcal{X} \to \mathbb{R}$ such that for an image region $x \in \mathcal{X}$, $d_k(x) > 0$ if the image region contains p_k and $d_k(x) \le 0$ otherwise. So, we need to learn K part detectors. A direct solution is to learn the K part detectors separately. One problem of this solution is that it ignores correlations among the parts. For example, according to the part definition in Section 3.1, if a part appears in an image region, any smaller part within this part is also visible. To exploit potential correlations among the parts, we propose a multi-label learning approach to learn the K part detectors jointly.

Specifically, we learn a function $F : \mathcal{X} \to 2^{\mathcal{P}}$ to predict a set of parts $P \subseteq \mathcal{P}$ which appear in a given image region x. Let $\mathcal{D} = \{(x_i, l_i, B_i^v, B_i^f) | 1 \le i \le N\}$ be a set of training examples, where $x_i \in \mathcal{X}$ is an image region, $l_i \in \{-1, 1\}$ indicates whether the image region contains a pedestrian and if so, B_i^v and B_i^f are two bounding boxes specifying the visible portion and full body of the pedestrian respectively. To learn F, we need to construct for each instance x_i a label vector $\boldsymbol{Y}_i = (y_{ik}) \in \{-1, 1\}^K$ for $1 \le k \le K$, where y_{ik} indicates whether the image region x_i contains the part p_k. When a part is only partially visible on a pedestrian, it is not trivial to assign 1 or -1 to the part. Wrong assignment of part labels may cause the learning of part detectors to fail. So, we introduce a cost vector $\boldsymbol{C}_i = (c_{ik}) \in \mathbb{R}^K$ for $1 \le k \le K$ to soften the label assignment, where c_{ik} $(0 \le c_{ik} \le 1)$ defines the cost incurred by a wrong prediction on x_i for p_k. For $l_i = -1$, we set $y_{ik} = -1$ and $c_{ik} = 1$. For $l_i = 1$, we set $y_{ik} = 1$ and determine the cost c_{ik} based on B_i^v and B_i^f. We first standardize the full-body bounding box B_i^f as in [6]: the height and center of B_i^f are fixed and its width is adjusted to make the ratio of the width to the height equal to 0.41. Denote by R_i^f the standardized full body. Then, any image contents inside the visible portion B_i^v but outside the standardized full body R_i^f are discarded to obtain a new visible portion R_i^v. Let R_{ik}^p be the image region of the part p_k on the instance x_i. We calculate the intersection over union (IOU) of R_{ik}^p and R_i^v denoted by I_{ik}, and the proportion of R_{ik}^p covered by R_i^v denoted by O_{ik}. Finally, the cost c_{ik} is determined as follows:

$$c_{ik} = \begin{cases} O_{ik} & O_{ik} \ge 0.7; \\ I_{ik} & O_{ik} < 0.7 \text{ and } I_{ik} \ge 0.5; \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

In the first case, the majority of the part p_k is visible on the instance x_i, so a large cost is set to prevent the part from being wrongly predicted. In the second case, the IOU of the part and vis-

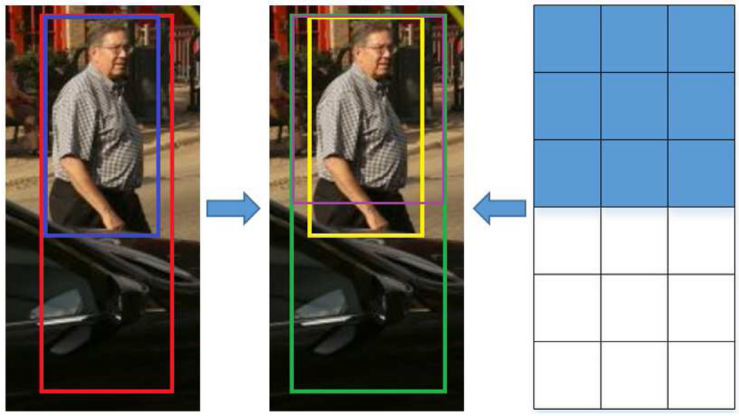


Fig. 3. Example labeling. The blue and red bounding boxes are the visible portion and full body of the pedestrian example respectively. The green bounding box is the standardized full body and the yellow bounding box is the new visible portion inside the standardized full body. The purple bounding box shows the image region of the upper-body part on the pedestrian example. The cost vector is calculated according to the purple and yellow bounding boxes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

ible portion is over 0.5, thus we consider the part to be visible on the instance and the cost of wrongly classifying it depends on the IOU. In the third case, the part is largely occluded. We set $c_{ik} = 0$ to ignore this training example for the k-th part.

Fig. 3 illustrates how a pedestrian example is labeled. $\mathcal{D}_F = \{(x_i, \boldsymbol{Y}_i, \boldsymbol{C}_i) | 1 \le i \le N\}$ forms the training set for learning F. We identify F with a two-argument function $H : \mathcal{X} \times \mathcal{P} \to \mathbb{R}$ such that $p_k \in F(x)$ if $H(x, p_k) > 0$ and $p_k \notin F(x)$ otherwise. For any predicate π, let $[\pi]$ be 1 if π holds and 0 otherwise. We learn H by minimizing the following loss function:

$$L(H, \mathcal{D}_F) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} c_{ik} [\text{sign}(H(x_i, p_k)) \neq y_{ik}], \tag{2}$$

where N is the number of training examples, $\text{sign}(H(x_i, p_k)) = 1$ if $H(x_i, p_k) > 0$ and $\text{sign}(H(x_i, p_k)) = -1$ otherwise.

3.3. Learning via boosting

Since decision trees when learned and combined via boosting have shown promising performance for pedestrian detection [2,3,15,16], we choose decision trees to form our part detectors. We explore two approaches to minimize the loss function $L(H, \mathcal{D}_F)$ in Eq. (2) for learning the K part detectors.

The first approach learns the K part detectors separately. Define the training loss related to the k-th part detector by

$$L_k(H, \mathcal{D}_F) = \frac{1}{N} \sum_{i=1}^{N} c_{ik} [\text{sign}(H(x_i, p_k)) \neq y_{ik}]. \tag{3}$$

$L(H, \mathcal{D}_F)$ can be decomposed as

$$L(H, \mathcal{D}_F) = \sum_{k=1}^{K} L_k(H, \mathcal{D}_F). \tag{4}$$

So, $L(H, \mathcal{D}_F)$ can be minimized by minimizing $L_k(H, \mathcal{D}_F)$ for $1 \le k \le K$ separately. Let $Q_k = \sum_{i=1}^{N} c_{ik}$. We normalize the costs associated with the k-th part by Q_k to obtain $\hat{\boldsymbol{C}}_k = (c_{1k}/Q_k, \ldots, c_{Nk}/Q_k)$. $\hat{\boldsymbol{C}}_k$ can be considered as a distribution over the training examples of the k-th part in $\mathcal{D}_F$. Boosting can be applied to learn and combine T decision trees to form a detector for the k-th part with example weights initialized to $\hat{\boldsymbol{C}}_k$. This learning approach has two limitations: (1) Correlations among the parts are ignored; (2) The computational costs of training and testing increase linearly with the number of parts.

To address the limitations of the separate learning approach, we propose another approach to learn the K part detectors jointly. Instead of learning T decision trees for each part detector, we only learn T decision trees which are shared by all the part detectors. We adapt AdaBoost.MH [17], which is a multi-class, multi-label version of AdaBoost, to learn H of the form:

$$H(x, p) = \sum_{t=1}^{T} \alpha_t h_t(x, p), \tag{5}$$

where $h_t : \mathcal{X} \times \mathcal{P} \to \mathbb{R}$ is a weak classifier which is a decision tree in our case and α_t is a weight associated with h_t. First, we consider a simplified case in which $c_{ik} = 1$ for $1 \le i \le N$ and $1 \le k \le K$. AdaBoost.MH can be directly applied to minimize $L(H, \mathcal{D}_F)$. The idea of AdaBoost.MH is to reduce the multi-label learning problem to a binary classification problem for which AdaBoost can be used to obtain H. Each training example $(x_i, \boldsymbol{Y}_i, \boldsymbol{C}_i) \in \mathcal{D}_F$ is replaced with K training examples $((x_i, p_k), y_{ik})$ for $1 \le k \le K$. Note that since $c_{ik} = 1$ for all i and k, $\boldsymbol{C}_i$ in the example $(x_i, \boldsymbol{Y}_i, \boldsymbol{C}_i)$ can be ignored. y_{ik} is the binary label for (x_i, p_k). $\mathcal{D}_B = \{((x_i, p_k), y_{ik}) | 1 \le i \le N, 1 \le k \le K\}$ forms the training set for the binary classification problem. H is learned by minimizing the following loss function:

$$L(H, \mathcal{D}_B) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \frac{1}{K} [\text{sign}(H(x_i, p_k)) \neq y_{ik}]. \tag{6}$$

AdaBoost.MH maintains a sequence of weight matrices $(\boldsymbol{W}^1, \ldots, \boldsymbol{W}^T)$ through T stages where $\boldsymbol{W}^t = (w_{ik}^t) \in \mathbb{R}^{N \times K}$ for $1 \le t \le T$ with w_{ik}^t the weight of the training example (x_i, p_k) at stage t. $\boldsymbol{W}^1$ is initialized to $w_{ik}^1 = \frac{1}{NK}$ for all i and k. At each stage t, AdaBoost.MH learns a weak classifier h_t and a weight α_t based on $\boldsymbol{W}^t$. With h_t, example weights are updated as follows:

$$w_{ik}^{t+1} = \frac{w_{ik}^t exp(-\alpha_t y_{ik} h_t(x_i, p_k))}{Z_t}, \tag{7}$$

where $Z_t = \sum_{i,k} w_{ik}^t exp(-\alpha_t y_{ik} h_t(x_i, p_k))$ is a normalization factor. The training error of the learned H is bounded by

$$L(H, \mathcal{D}_F) \le \prod_{t=1}^{T} Z_t. \tag{8}$$

Now we introduce how to minimize $L(H, \mathcal{D}_F)$ for the general case. Note that when $\frac{1}{K}$ in Eq. (6) is replaced with c_{ik}, the loss function in Eq. (6) is exactly the loss function in Eq. (2). It is easy to verify that by initializing $\boldsymbol{W}^1$ to $w_{ik}^1 = \frac{c_{ik}}{N}$ for all i and k, AdaBoost.MH can be used to minimize $L(H, \mathcal{D}_F)$ in Eq. (2). The upper loss bound given in Eq. (8) still holds.

Next, we describe how to learn a decision tree h_t at stage t given example weights $\boldsymbol{W}^t$. Starting with a root node, we construct h_t greedily. At the beginning, all training examples fall in the root node with sample weights $\boldsymbol{W}^t$. We examine one leaf node at a time. If a leaf node reaches a predefined maximum depth or the training examples falling in the node are pure enough, we stop branching the leaf node. Otherwise, we choose a feature and a threshold which have the minimum weighted classification error on the training examples reaching the leaf node. With the chosen feature and threshold, the training examples are split into two subsets each of which is assigned to one new leaf node. The two new leaf nodes are the children of the current leaf node. Assume h_t has M leaf nodes and the instance space $\mathcal{X}$ is partitioned into X_1, ... ,X_M with X_j the set of instances falling in the j-th leaf node. For an instance $x \in X_j$, h_t is defined to output

$$h_t(x, p_k) = \frac{1}{2}\ln\left(\frac{S^+_{jk}}{S^-_{jk}}\right), \tag{9}$$

where $S^+_{jk} = \sum_{x_i \in X_j} w^t_{ik}[y_{ik} = 1]$ and $S^-_{jk} = \sum_{x_i \in X_j} w^t_{ik}[y_{ik} = -1]$. After the decision tree is constructed, it can be proved that h_t defined in Eq. (9) minimizes Z_t with α_t set to 1 (See [17] for more details).

According to the above adaptation of AdaBoost.MH for minimizing $L(H, \mathcal{D}_F)$, the costs $\boldsymbol{C} = (c_{ik}) \in \mathbb{R}^{N \times K}$ after normalization can be considered as a distribution over $\mathcal{D}_B$. The decision trees are learned to capture the overall distribution of all the parts. Part correlations are exploited by sharing the decision trees among these part detectors. When taken as a pedestrian detector individually, the part detectors learned jointly show better performance than their counterparts learned separately as demonstrated in Section 5. For detection, applying the part detectors with shared decision trees is much faster as it only involves a computational cost of T instead of $K \times T$ decision trees.

3.4. Complexity analysis

In general, when part detectors are learned separately, the computational complexity of applying these part detectors for detection is $O(K \times C)$, where K and C are the number of part detectors and the cost of applying each part detector respectively. In our approach, T decision trees are shared by all the K part detectors and each leaf in a decision tree stores K responses. Assume the maximum depth of each decision tree is D. The computation cost of applying one decision tree is $D - 1$ comparisons $+ K$ additions. So, the total computational complexity is $T \times (D - 1 + K)$ operations or $O(T \times (D + K))$. When the part detectors are learned separately, each part detector consists of T decision trees. The computational cost of applying one part detector is $O(T \times D)$. So, the total computational complexity is $O(K \times T \times D)$. Compared with separate learning, our approach achieves a speedup of $O((K \times D)/(D + K))$.

4. Occlusion handling with part detectors

4.1. Integrating part detectors

In a particular scene, pedestrians may be occluded by each other or other objects. Simply applying a full body detector usually does not work well when pedestrians are heavily occluded. As we do not know in advance which parts are occluded, a simple yet effective way to handle occlusions is to apply a set of part detectors. For a candidate region x in an image, the K part detectors would give K detection scores, $\boldsymbol{s} = (s_1, \ldots, s_K)$, where $s_k = H(x, p_k)$ for $1 \le k \le K$. We need to integrate these detection scores properly to give a final score indicating how likely the candidate region contains a pedestrian. We study three methods for this purpose.

Let $g : \mathbb{R}^K \to \mathbb{R}$ be an integration function. A common integration method is to take the maximum among the K detection scores:

$$g(\boldsymbol{s}) = \max_{1 \le k \le K} s_k. \tag{10}$$

We call this method Max integration.

The second method we adopt is a variant of Max integration which takes the average of the top S detection scores with $1 \le S \le K$. Let $\boldsymbol{s}'$ be a vector obtained by sorting the detection scores in $\boldsymbol{s}$ in descending order. The integration function can be expressed as

$$g(\boldsymbol{s}) = \frac{1}{S}\sum_{1 \le k \le S} s'_k. \tag{11}$$

We call this method TopS integration. This method is based on two observations as illustrated in Fig. 4: (1) for a partially occluded pedestrian, detectors of those parts which are inside or have a large overlap with the visible region of the pedestrian would probably give high detection scores, while the other part detectors may give low detection scores due to occlusions; (2) for a non-pedestrian region, part detectors tend to output low detection scores.

The third method learns a weight vector $\boldsymbol{v} = (v_1, \ldots, v_K)$ to linearly combine the K detection scores:

$$g(\boldsymbol{s}) = \sum_{1 \le k \le K} v_k s_k + b. \tag{12}$$

where b is a bias term. Let $\mathcal{D} = \{(\boldsymbol{s}_i, y_i) | 1 \le i \le N\}$ be a set of training examples, where $\boldsymbol{s}_i$ represents the K detection scores of the i-th training example and $y_i \in \{-1, 1\}$. We learn $\boldsymbol{v}$ by logistic regression. Specifically, $\boldsymbol{v}$ is learned by solving the following optimization problem:

$$\min_{\mathbf{v}} \phi(\mathbf{v}) + \lambda \sum_{i=1}^{N} \log\left(1 + exp(-y_i g(\boldsymbol{s}_i))\right), \tag{13}$$

where N is the number of training examples and $\phi(\mathbf{v})$ is a regularization term. Following [13], we adopt two types of regularizers, i.e. $\phi(\mathbf{v}) = \frac{1}{2}\mathbf{v}^T\mathbf{v}$ and $\phi(\mathbf{v}) = ||\mathbf{v}||_1$. When the L1 norm $||\mathbf{v}||_1$ is used, a subset of part detectors are selected with a proper choice of λ. We call this method L1 integration if the L1 norm is used in Eq. (13) and otherwise L2 integration.

4.2. Contextual rescoring

Context is commonly used to improve performance for object detection [40,46,49]. We exploit context based on the observation that for our part detectors, detections are densely distributed around a true detection while rare detections are found around a false detection as illustrated in Fig. 5. For each detection d, we collect M top-scoring neighboring detections whose intersection over union (IOU) with d is greater than 0.5. Let q be the score for the detection d and q^{nb}_i be the scores for the i-th neighboring detection. These scores come from a specific integration method, e.g. L2 integration. The scores q^{nb}_i $(1 \le i \le M)$ are sorted in descending order. We represent the detection d by a score vector $\boldsymbol{u} = (q, q^{\text{nb}}_1, \ldots, q^{\text{nb}}_M)$ with $q^{\text{nb}}_1 \ge \ldots \ge q^{\text{nb}}_M$. If the detection d only has less than M neighboring detections, the remaining entries in $\boldsymbol{u}$ are filled with 0. The score vector of a true detection tends to have more high scores than the score vector of a false detection, which can be exploited to further distinguish between true and false detections. To do this, we learn a rescoring model based on the above representation using a non-linear support vector machine (SVM)

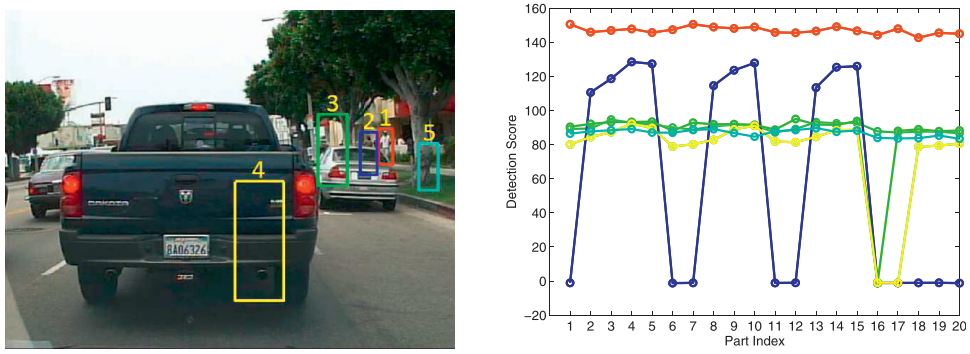


Fig. 4. Part scores on top 5 scoring image regions. The number above each region denotes its ranking. For the fully visible pedestrian (Red bounding box), all part detectors give high detection scores consistently (Red curve). For the partially occluded pedestrian (Blue bounding box), only the detectors of visible parts (e.g. P3, P4, P5, P9, P10, P14 and P15) output high detection scores (Blue curve). Background regions (Green, yellow and cyan bounding boxes) receive relatively low scores from all the part detectors (Green, yellow and cyan curves). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

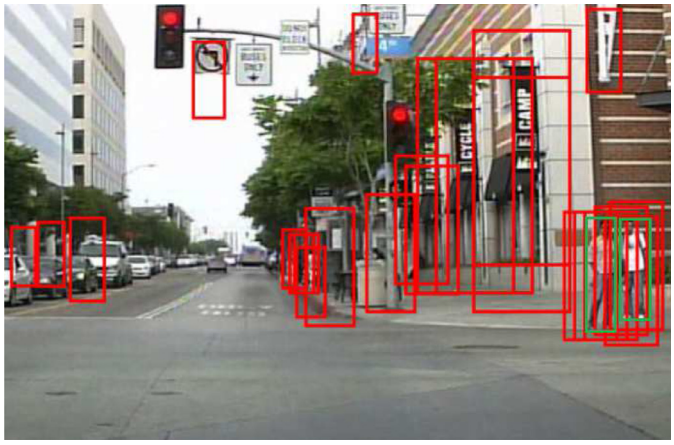


Fig. 5. Detection distribution. Red bounding boxes are thirty detections with the highest scores. Detections are densely located around the two pedestrians marked by green bounding boxes. Detections are sparsely distributed in other regions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

with an intersection kernel. Specifically, we learn a rescoring function of the following form:

$$r(\boldsymbol{u}) = \sum_{i=1}^{N} \alpha_i \kappa(\boldsymbol{u}, \boldsymbol{u}_i), \tag{14}$$

where N is the number of training examples, $\boldsymbol{u}_i$ is the i-th training example, α_i for $1 \le i \le N$ are the parameters to be learned and

$$\kappa(\boldsymbol{u}, \boldsymbol{u}_i) = \sum_{j=1}^{M+1} \min(\boldsymbol{u}(j), \boldsymbol{u}_i(j)), \tag{15}$$

with $\boldsymbol{u}(j)$ the j-th entry in $\boldsymbol{u}$. To accelerate training and testing, we approximate the intersection kernel by a linear kernel

$$\kappa(\boldsymbol{u}, \boldsymbol{u}_i) \approx \boldsymbol{u}'^T \boldsymbol{u}'_i, \tag{16}$$

where $\boldsymbol{u}'$ and $\boldsymbol{u}'_i$ are approximate explicit mappings of $\boldsymbol{u}$ and $\boldsymbol{u}_i$ respectively obtained by the technique proposed in [53]. Then, we learn the rescoring model by a linear SVM.

5. Experiments

We apply our approach to two types of features, hand-crafted channel features and features learned by a convolutional neural network (CNN features), and evaluate its effectiveness on two pedestrian detection datasets, Caltech [6] and CUHK [9].

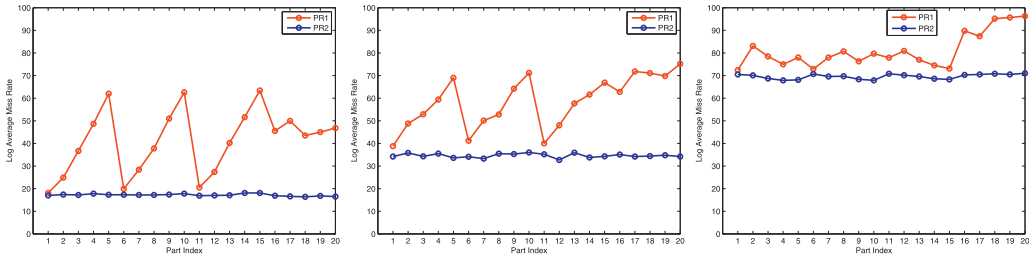
The Caltech dataset is commonly used for evaluating pedestrian detection approaches and provides both visible portion and full body annotations. Following the standard evaluation protocol, we use video sets S0-S5 for training and video sets S6-S10 for testing. A log-average miss rate, which is calculated by averaging miss rates at 9 false positive per-image (FPPI) points sampled evenly in the log-space ranging from 10^{-2} to 10^{0}, is used to summarize the detection performance. As the purpose of our approach is to handle occlusions, we evaluate it on three subsets: *Reasonable, Partial* and *Heavy*. In the *Reasonable* subset, only pedestrians with at least 50 pixels tall and under no or partial occlusion are used for evaluation. This subset is widely used for evaluating pedestrian detection approaches. In the *Partial* and *Heavy* subsets, pedestrians are at least 50 pixels tall and are partially occluded (1–35% occluded) and heavily occluded (36–80% occluded) respectively.

The CUHK dataset is specially collected for evaluating occlusion handling approaches for pedestrian detection. The dataset consists of 1063 images from Caltech, ETHZ, TUD-Brussels, INRIA, Caviar and other sources. In this dataset, each image contains at least one partially occluded pedestrian. As this dataset only contains a small number of images, we do not train detectors on this dataset. Instead, we use this dataset to evaluate the generalizability of detectors trained on the Caltech dataset. Similar to the Caltech dataset, evaluation is done on the *Reasonable, Partial* and *Heavy* subsets, and detection performance is summarized by the log-average miss rate.

5.1. Experiments with channel features

5.1.1. Implementation

We choose locally decorrelated channel features (LDCF) [18] which are frequently used for pedestrian detection in recent years to represent the parts in our approach. We use the same setting as in [18]: 4 filters of size 5×5 are learned to locally decorrelate aggregated channel features (ACF) [15] of 10 channels to generate LDCF of 40 channels. We sample training data from video sets S0-S5 at an interval of 3 frames. Pedestrian examples which are at least 50 pixels tall and occluded not more than 70% are collected as positive examples. Five rounds of bootstrapping are adopted to train 64, 512, 1024, 2048 and 4096 decision trees respectively. The maximum depth of a decision tree is 5. We use $S = 15$ for the TopS integration method. $C = 0.1$ and $C = 0.001$ are used for L2 integration and L1 integration respectively.



(a) *Reasonable* (b) *Partial* (c) *Heavy*

Fig. 6. Results of part detectors using different part representations on the Caltech dataset.

Table 1
Comparison of separate learning (SL) and joint learning (JL) using channel features on the Caltech dataset. P1-P4, P6 and P11 are six typical parts shown in Fig. 2. The last row shows the average improvements on the three subsets brought by joint learning.

Method	*Reasonable*	*Partial*	*Heavy*
SL-P1	18.2	36.1	72.1
JL-P1	**17.0**	**34.2**	**70.5**
SL-P2	19.3	40.2	71.9
JL-P2	**17.4**	**35.8**	**70.1**
SL-P3	18.6	39.4	69.4
JL-P3	**17.2**	**34.3**	**68.7**
SL-P4	18.6	39.7	69.9
JL-P4	**17.8**	**35.5**	**67.9**
SL-P6	19.2	37.7	72.1
JL-P6	**17.3**	**34.1**	**70.7**
SL-P11	19.2	42.4	73.8
JL-P11	**16.9**	**35.2**	**70.8**
Avg. Imp.	+1.6	+4.4	+1.8

Table 2
Comparison of cost-sensitive labeling and hard learning using channel features on the Caltech dataset. P1–P4, P6 and P11 are six typical parts shown in Fig. 2. The last row shows the average improvements on the three subsets brought by cost-sensitive labeling.

Method	*Reasonable*	*Partial*	*Heavy*
HL-P1	20.9	44.2	79.5
JL-P1	**17.0**	**34.2**	**70.5**
HL-P2	22.1	43.7	77.9
JL-P2	**17.4**	**35.8**	**70.1**
HL-P3	24.5	46.0	74.1
JL-P3	**17.2**	**34.3**	**68.7**
HL-P4	31.4	53.2	73.6
JL-P4	**17.8**	**35.5**	**67.9**
HL-P6	22.6	44.0	82.0
JL-P6	**17.3**	**34.1**	**70.7**
HL-P11	22.4	46.3	83.0
JL-P11	**16.9**	**35.2**	**70.8**
Avg. Imp.	+6.7	+11.4	+8.6

Table 3
Results of full-body detectors and integration methods using channel features on the Caltech dataset.

Method	*Reasonable*	*Partial*	*Heavy*
LDCF-P1	18.1	38.8	72.4
SL-P1	18.2	36.1	72.1
HL-P1	20.9	44.2	79.5
JL-P1	17.0	34.2	70.5
JL-Max	17.5	34.8	**68.8**
JL-TopS	16.6	**32.7**	69.6
JL-L1	16.7	33.1	69.0
JL-L2	**16.5**	33.0	68.9

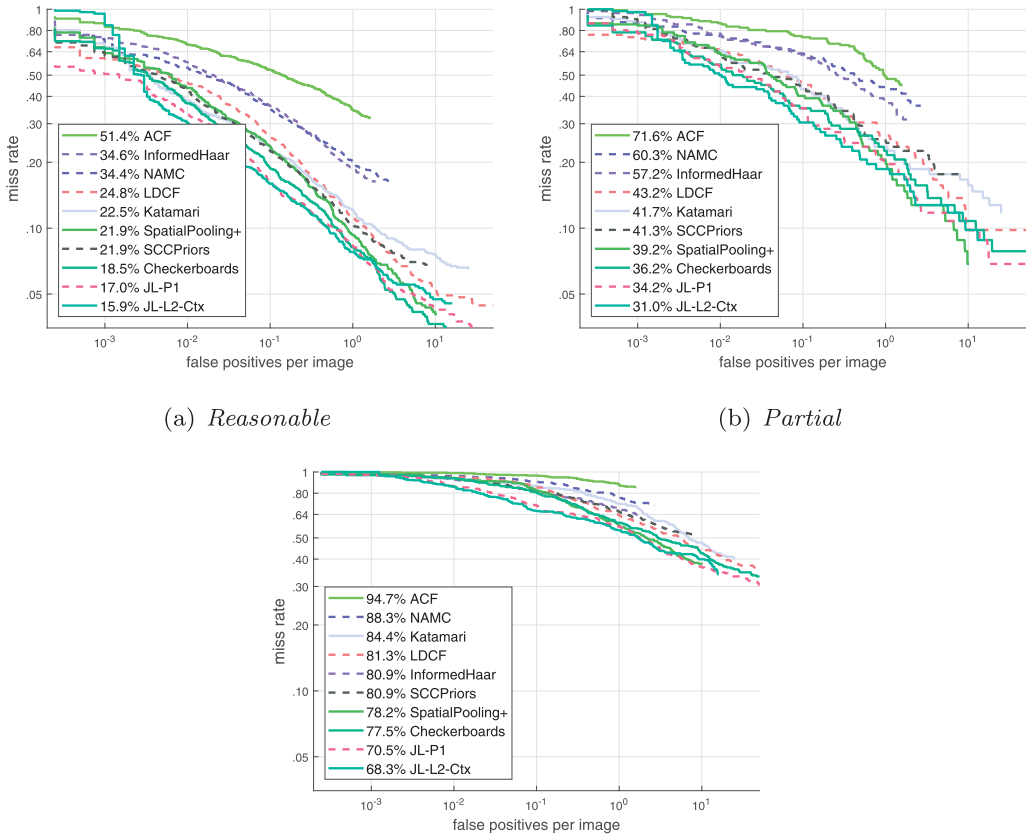
5.1.2. Experimental results on Caltech

Fig. 6 shows the results of part detectors learned using different part representations. PR1 denotes the representation method in which a part is represented by the features from its own image region. The part detectors using PR1 are learned independently. PR2 is the representation method in which all the parts share the features from the whole body. The part detectors using PR2 are learned jointly using our multi-label formulation. It can be seen that the performances of the part detectors learned using PR1 vary largely from part to part on the *Reasonable*, *Partial* and *Heavy* subsets. The detectors of small parts (e.g. P5, P10 and P20) usually perform worse than those of large parts (e.g. P1, P6 and P11) since with PR1, the information from the small parts is relatively limited compared with that from the large parts (See Fig. 2 for part correspondence). The part detectors using PR2 perform much better than those using PR1. The performances of different part detectors using PR2 do not change much on the three subsets. Although these part detectors show similar performances, they do behave differently. The example distribution of each part is captured by its detector. When a pedestrian example is occluded, those parts inside or have large overlap with the visible portion usually get large detection scores while the other parts tend to have low detection scores (See the blue curve in Fig. 4).

Table 1 shows the results of the detectors of six typical parts (P1-P4, P6 and P11) learned by two different approaches, separate learning (SL) and joint learning (JL). SL learns part detectors separately by minimizing Eq. (3), while JL learns all part detectors jointly by minimizing Eq. (2) using AdaBoost.MH. For the six parts, the detectors learned by JL perform better than their counterparts learned by SL on all the three subsets, which shows the effectiveness of sharing decision trees to exploit correlations among the parts. The average improvements on the three subsets brought by JL are 1.6%, 4.4% , and 1.8% respectively.

Table 2 compares two labeling strategies: cost-sensitive labeling and hard labeling. The proposed joint learning approach adopts a cost-sensitive labeling strategy: besides a class label, each training example is also associated with a misclassification cost (See Section 3.2). To demonstrate the effectiveness of the cost-sensitive labeling strategy, we consider a hard labeling strategy in which every training example has a misclassification cost of 1. JL is our approach with cost-sensitive labeling and HL also learns the part detectors jointly by minimizing Eq. (2) but with the hard labeling strategy. The detectors of six typical parts (P1–P4, P6 and P11) learned by JL perform significantly better than their counterparts learned by HL with mean improvements of 6.7%, 11.4% and 8.6% on the *Reasonable*, *Partial* and *Heavy* subsets respectively. From the comparison, we can see that proper labeling of training examples is important for our multi-label learning approach to work well.

Table 3 shows the results of different full-body detectors and integration methods. LDCF-P1 is our implementation of LDCF [18] which only uses fully visible pedestrian examples as positive



(a) *Reasonable* (b) *Partial*

(c) *Heavy*

Fig. 7. Comparison with state-of-the-art using channel features on the Caltech dataset.

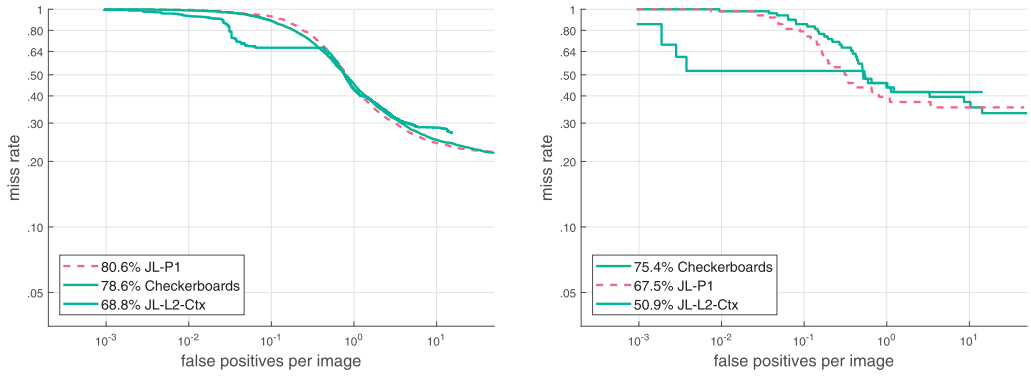
examples. LDCF-P1 and SL-P1 performs closely on the *Reasonable* and *Heavy* subsets, but SL-P1 outperforms LDCF-P1 by 2.7% on the *Partial* subset since SL-P1 uses additional partially occluded pedestrian examples for training according to the definition of the misclassification cost in Eq. (1). JL-P1 achieves the best performance among the four full-body detectors on all the three subsets. Particularly, JL-P1 outperforms SL-P1 by 1.2%, 1.9% and 1.6% on the three subsets respectively. JL-Max, JL-TopS, JL-L1 and JL-L2 represent the four integration methods (Max, TopS, L1 and L2 described in Section 4.1) applied to the 20 part detectors learned by JL. Except JL-Max, the other three integration methods outperform JL-P1 on all the three subsets, which demonstrates that properly integrating part detectors can improve the performance in different occlusion situations. JL-TopS performs slightly worse than JL-Max on the *Heavy* subset, but outperforms it by 0.9% and 2.1% on the *Reasonable* and *Partial* subsets respectively. Overall, TopS integration is more robust than Max integration. JL-TopS, JL-L1 and JL-L2 perform closely on the three subsets. Particularly, JL-L1 does not show advantage over JL-L2, which is different from the results reported in [13]. This is because the part detectors used in [13] are learned independently using PR1 while JL-L1 and JL-L2 uses the part detectors learned jointly using PR2. The performances of the detectors learned using PR1 vary largely while the part detectors learned jointly using PR2 are more robust as illustrated in Fig. 6. In [13], the L1 regularization can remove unstable detectors to eliminate the negative effects of these detectors. For JL-L1 and JL-L2, all the part detectors show stable performances and it is not necessary to remove any part detectors. Contrarily, JL-L2 shows slightly better performance than JL-L1 on the three subsets. In addition, the advantage of JL-L2 over JL-Max is not as significant as that reported in [13]. This is because the part detectors used in JL-L2 are learned jointly, which makes their outputs more comparable.

Table 4
Results of approaches with different part pools on the Caltech dataset.

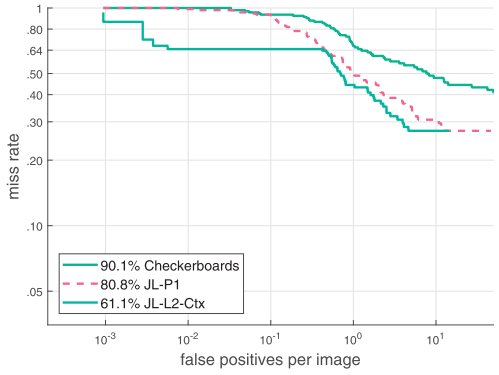
Part pool	Method	*Reasonable*	*Partial*	*Heavy*
PP10	JL-P1	17.5	34.6	70.2
	JL-L2	17.3	34.5	70.0
	JL-L2-Ctx	17.3	33.7	69.3
PP20	JL-P1	17.0	34.2	70.5
	JL-L2	16.5	33.0	68.9
	JL-L2-Ctx	**15.9**	**31.0**	68.3

Table 4 shows the results of three approaches with two different part pools. JL-L2-Ctx is an approach which combines JL-L2 and the contextual rescoring model described in Section 4.2. PP20 = {P1–P20} is the part pool defined in Fig. 2. PP10 = {P1–P6, P11, P16–P18} is a subset of PP20. For both part pools, JL-L2 performs better than JL-P1 and JL-L2-Ctx performs best on the three subsets, which shows that both L2 integration and contextual rescoring contribute to performance improvement. These approaches work better with PP20 since a larger part pool can better cover occlusion situations in the dataset.

Fig. 7 gives a comparison of our approach and state-of-the-art approaches using channel features, ACF [15], InformedHaar [29], NAMC [54], LDCF [18], Katamari [20], SpatialPooling+ [1], SCCPriors [55] and Checkerboards [2]. Our approach achieves the best performance among these channel-feature based approaches. Our full-body detector (JL-P1) already outperforms Checkerboards on all the three subsets. The improvement of JL-P1 over Checkerboards is significant (7%) on the *Heavy* subset. With L2 integration



(a) *Reasonable* (b) *Partial*



(c) *Heavy*

Fig. 8. Results using channel features on the CUHK dataset.

Table 5
Results of full-body detectors and integration methods using CNN features on the Caltech dataset.

Method	*Reasonable*	*Partial*	*Heavy*
RPN+BF-P1	10.1	18.9	58.9
SL-P1	10.3	18.0	56.6
HL-P1	10.8	22.9	66.6
JL-P1	**9.9**	17.2	50.5
JL-Max	10.3	17.2	**48.4**
JL-TopS	10.0	16.6	49.2
JL-L1	10.0	**16.5**	49.3
JL-L2	10.0	16.6	49.2

and contextual rescoring, JL-L2-Ctx outperforms JL-P1 on the three subsets by 1.1%, 3.2% and 2.2% respectively. Both JL-P1 and JL-L2-Ctx outperform Checkerboards significantly (7% and 9.2% respectively) on the *Heavy* subset, which demonstrates the effectiveness of our approach for detecting heavily occluded pedestrians.

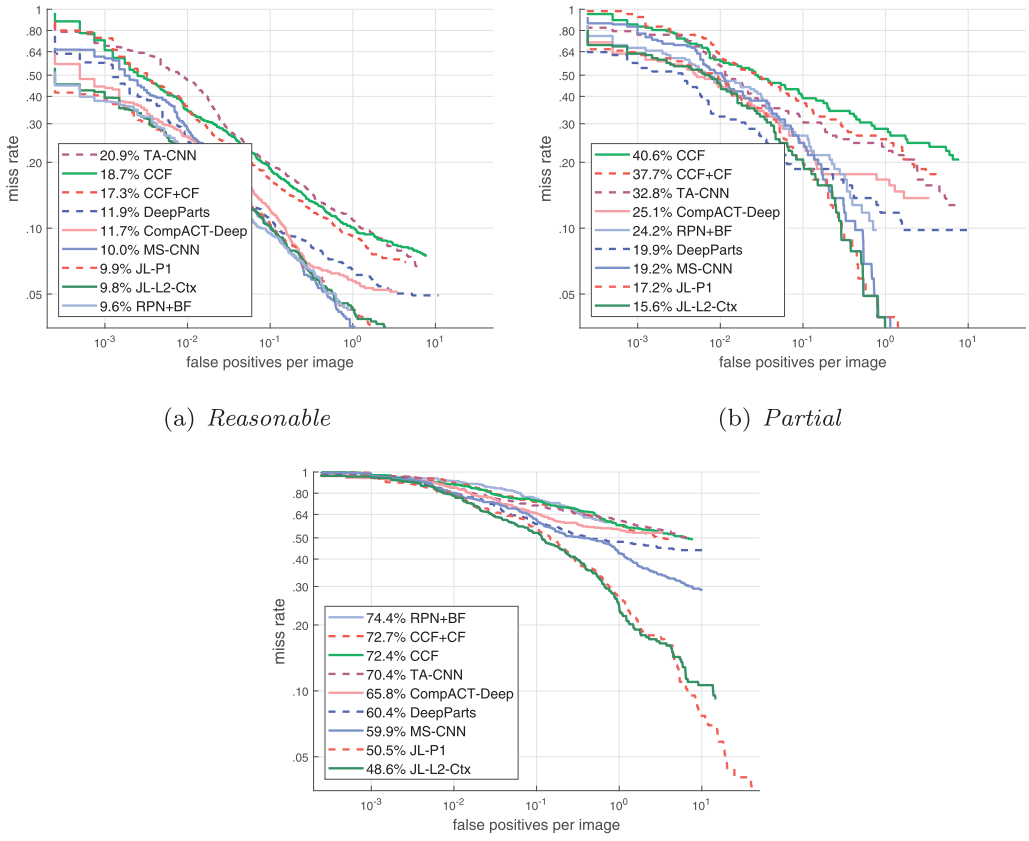
5.1.3. Experimental results on CUHK

We compare our approach with Checkerboards [2] on this dataset. Fig. 8 shows the results of Checkerboards, JL-P1 and JL-L2-Ctx. For Checkerboards, we use the code and its model trained on the Caltech dataset which are publicly available to generate the result. JL-P1 performs 2% worse than Checkerboards on the *Reasonable* subset but outperforms Checkerboards by 7.9% and 9.3% on the *Partial* and *Heavy* subsets respectively, which shows the advantage of learning the full-body detector jointly with other part detectors to exploit part correlations. With L2 integration and contextual rescoring, JL-L2-Ctx outperforms JL-P1 by 11.8%, 16.6% and 19.7% on the *Reasonable*, *Partial* and *Heavy* subsets respectively. JL-L2-Ctx shows significant advantage over Checkerboards on all the three subsets.

5.2. Experiments with CNN features

5.2.1. Implementation

Recently, several approaches using CNN features have achieved the state-of-the-art performance for pedestrian detection [3,4,14,33]. The proposed multi-label learning approach also applies to CNN features. We use a region proposal network (RPN) from [3] for feature extraction and then learn a set of part detectors jointly as described in Section 3.3. RPN+BF [3] also adopts a similar framework in which a set of decision trees are learned to form a full-body detector using CNN features from the RPN. The major differences between RPN+BF and our approach are two-fold: (1) our approach jointly learns the full-body detector with the other part detectors to exploit part correlations; (2) our approach further integrates the part detectors to better handle occlusions. We sample training data from video sets S0-S5 at an interval of 3 frames as in [3]. Pedestrian examples which are at least 50 pixels tall and occluded not more than 70% are collected as positive examples. These positive examples are also used for training the RPN (See [3] for the network architecture and training procedure of the PRN). To speed up training and testing, we use the RPN to generate pedestrian proposals. About 1000 proposals and 400 proposals per image are generated for training and testing respectively. Six rounds of bootstrapping are adopted to train 64, 128, 256, 512, 1024 and 2048 decision trees respectively. The maximum depth of a decision tree is 5. We use $S = 15$ for the



(a) *Reasonable* (b) *Partial*

(c) *Heavy*

Fig. 9. Comparison with state-of-the-art using CNN features on the Caltech dataset.

TopS integration method. $C = 0.1$ and $C = 0.005$ are used for L2 integration and L1 integration respectively.

5.2.2. Detection speed

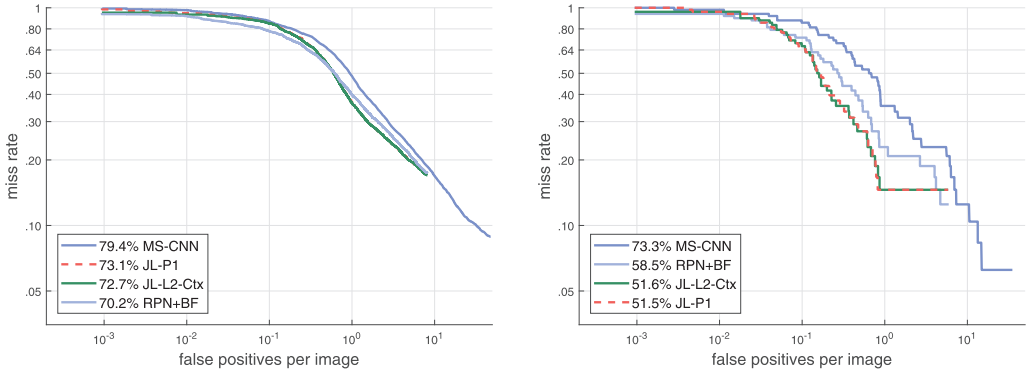
On an NVIDIA K5200 GPU, it takes about 0.6 s (0.5 s for feature extraction and 0.1 s for detection) to test the jointly learned part detectors on a 480×640 image, while it takes about 2.2 s (0.5 s + 1.7 s) to apply 20 separately learned detectors. Excluding the time for feature extraction, the speedup factor of the jointly learned part detectors is close to $20\times$.

5.2.3. Experimental results on Caltech

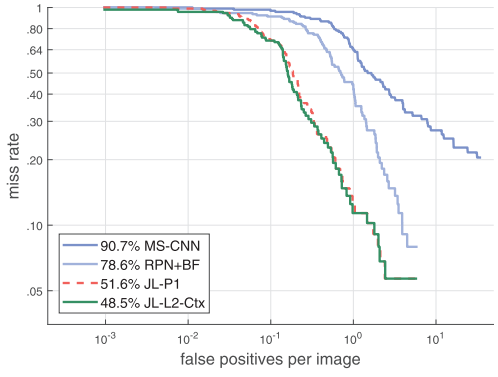
Table 5 shows the results of different full-body detectors and integration methods using CNN features. RPN+BF-P1 is our implementation of RPN+BF [3]. SL-P1 and JL-P1 are two full-body detectors learned by separate learning (SL) and joint learning (JL) respectively. SL-P1 outperforms slightly worse than RPN+BF-P1 on the *Reasonable* subset but outperforms it on the *Partial* and *Heavy* subsets. The use of some partially occluded pedestrian examples for training makes SL-P1 achieve better performance for occluded pedestrian detection. JL-P1 outperforms SL-P1 on the three subsets by 0.4% (*Reasonable*), 0.8% (*Partial*) and 6.1% (*Heavy*) respectively. The performance improvement on *Heavy* is significant. In our multi-label learning approach, the full-body detector (JL-P1) is learned jointly with the other part detectors by sharing decision trees. These decision trees are learned to capture the overall distribution of pedestrian examples including heavily occluded ones. When the full-body detector is learned separately, most heavily occluded pedestrian examples are ignored, which makes SL-P1 perform relatively poorly on the *Heavy* subset. Using CNN features, HL-P1 performs much worse than JL-P1, which is consistent with the case of channel features. JL-Max, JL-TopS, JL-L1 and JL-L2 are four methods which use Max, TopS, L1 and L2 respectively to integrate the 20 part detectors learned by JL. The four integration methods show better performance than the single full-body detector JL-P1 on the *Partial* and *Heavy* subsets, which justifies that properly integrating part detectors can better handle occlusions than a single full-body detector. On the *Reasonable* subset, all the four integration methods perform slightly worse than JL-P1. Since JL-P1 already works well for detecting pedestrians which are non-occluded or slightly occluded, integrating the other part detectors with the full-body detector does not help. JL-Max has better performance than JL-TopS on the *Heavy* subset, while JL-TopS outperforms JL-Max on the *Reasonable* and *Partial* subsets. JL-TopS, JL-L1 and JL-L2 have similar performances.

Fig. 9 gives a comparison of our approach and some state-of-the-art CNN-feature based approaches, TA-CNN [32], CCF [31], CCF+CF [31], DeepParts [14], CompACT-Deep [4], MS-CNN [33] and RPN+BF [3]. On the *Reasonable* subset, JL-P1 performs comparably to the top two approaches RPN+BF and MS-CNN which also only use a single full-body detector. This is because the three approaches use similar deep convolutional neural networks (variants of VGG-16 [56]). On the *Partial* and *heavy* subsets, JL-P1 outperforms the most competitive approach MS-CNN by 2.0% and 9.4% respectively. The advantage of JL-P1 over MS-CNN on *Heavy* is significant, which shows the effectiveness of learning the full-body detector jointly with the other part detectors. With L2 integration and contextual rescoring, JL-L2-Ctx further improves the performance over JL-P1, especially on the *Partial* and *Heavy* subsets. JL-L2-Ctx outperforms MS-CNN by 0.2%, 3.6% and 11.3% on the *Rea-*

Fig. 10. Detection examples of our approach on the Caltech dataset.



(a) *Reasonable* (b) *Partial*



(c) *Heavy*

Fig. 11. Results on CUHK using CNN features.

sonable, *Partial* and *Heavy* subsets respectively. Fig. 10 shows some detection examples of our approach on the Caltech dataset.

5.2.4. Experimental results on CUHK

We compare our approach with RPN+BF [3] and MS-CNN [33] on this dataset. For RPN+BF and MS-CNN, we use the published models trained on the Caltech dataset. Fig. 11 shows the results of RPN+BF, MS-CNN, JL-P1 and JL-L2-Ctx. MS-CNN performs poorly on this dataset. JL-P1 performs 2.9% worse than RPN+BF on the *Reasonable* subset, but outperforms it by 7% and 27% on the *Partial* and *Heavy* subsets, which demonstrates the effectiveness of our multi-label learning approach. With linear integration and contextual rescoring, JL-L2-Ctx achieves better performance than JL-P1. Fig. 12 shows some detection examples of our approach on the CUHK dataset.

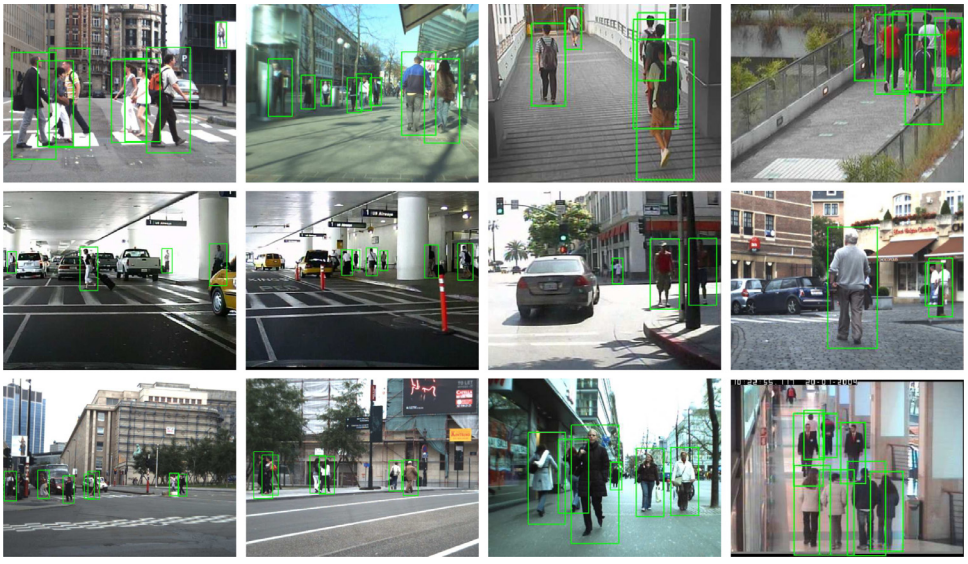


Fig. 12. Detection examples of our approach on the CUHK dataset.

6. Conclusion

In this paper, we propose a multi-label learning approach to learn part detectors jointly. AdaBoost.MH is adapted to learn a set of decision trees which are shared by all the part detectors. Thanks to the sharing of decision trees, part correlations are exploited and the computational cost of applying these part detectors is reduced. The learned decision trees capture the overall distribution of all the parts. We explore several methods to integrate the part detectors learned by the proposed approach for occlusion handling. We also proposes a contextual rescoring model to further improve the performance. The proposed approach is applied to hand-crafted channel features and CNN features respectively. Its effectiveness is validated on the Caltech and CUHK datasets. The experimental results show that (1) the part detectors learned jointly by the proposed approach perform better than their counterparts learned separately; (2) proper integration of these part detectors can improve the performance for detecting occluded pedestrians, especially heavily occluded ones; (3) contextual rescoring can further improve the performance for pedestrian detection. Currently, we use features from a deep CNN to construct a set of decision trees as part detectors which are then integrated to handle occlusions for pedestrian detection. Training and integration of part detectors are done in two separate steps. In future work, we will explore how to train the set of part detectors as well as integrate them in a single deep CNN.

Acknowledgment

This work is supported in part by Singapore Ministry of Education Academic Research Fund Tier 2 MOE2015-T2-2-114 and start-up grants of University at Buffalo.

References

[1] S. Paisitkriangkrai, C. Shen, A. Hengel, Strengthening the effectiveness of pedestrian detection with spatially pooled features, in: European Conference on Computer Vision (ECCV), 2014, pp. 546–561.
[2] S. Zhang, R. Benenson, B. Schiele, Filtered channel features for pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1751–1760.
[3] L. Zhang, L. Lin, X. Liang, K. He, Is faster r-cnn doing well for pedestrian detection? in: European Conference on Computer Vision (ECCV), 2016, pp. 443–457.
[4] Z. Cai, M. Saberian, N. Vasconcelos, Learning complexity-aware cascades for deep pedestrian detection, in: International Conference on Computer Vision (ICCV), 2015, pp. 3361–3369.
[5] D. Xu, W. Ouyang, E. Ricci, X. Wang, N. Sebe, Learning cross-modal deep representations for robust pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4236–4244.
[6] P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: an evaluation of the state of the art, IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 34 (2012) 743–761.
[7] M. Enzweiler, A. Eigenstetter, B. Schiele, D. Gavrila, Multi-cue pedestrian classification with partial occlusion handling, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 990–997.
[8] G. Duan, H. Ai, S. Lao, A structural filter approach to human detection, in: European Conference on Computer Vision (ECCV), 2010, pp. 238–251.
[9] W. Ouyang, X. Wang, A discriminative deep model for pedestrian detection with occlusion handling, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3258–3265.
[10] M. Mathias, R. Benenson, R. Timofte, L. Van Gool, Handling occlusions with Franken-classfiers, in: International Conference on Computer Vision (ICCV), 2013, pp. 1505–1512.
[11] W. Ouyang, X. Zeng, X. Wang, Modeling mutual visibility relationship in pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 3222–3229.
[12] W. Ouyang, X. Wang, Joint deep learning for pedestrian detection, in: International Conference on Computer Vision (ICCV), 2013, pp. 2056–2063.
[13] C. Zhou, J. Yuan, Learning to integrate occlusion-specific detectors for heavily occluded pedestrian detection, in: Asian Conference on Computer Vision (ACCV), 2016, pp. 305–320.
[14] Y. Tian, P. Luo, X. Wang, X. Tang, Deep learning strong parts for pedestrian detection, in: International Conference on Computer Vision (ICCV), 2015, pp. 1904–1912.
[15] P. Dollar, R. Appel, S. Belongie, P. Perona, Fast feature pyramids for object detection, IEEE Trans. Pattern Anal. Mach.Intell 36 (2014) 1532–1545.
[16] A. Costea, S. Nedevschi, Semantic channels for fast pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2360–2368.
[17] R. Schapire, Y. Singer, Improved boosting algorithms using confidence-rated predictions, J. R. Stat. Soc. Ser. B 37 (1999) 297–336.
[18] W. Nam, P. Dollar, J. Han, Local decorrelation for improved pedestrian detection, in: Advances in Neural Information Processing Systems (NIPS), 2014, pp. 424–432.
[19] C. Zhou, J. Yuan, Multi-label learning of part detectors for heavily occluded pedestrian detection, in: International Conference on Computer Vision (ICCV), 2017, pp. 3506–3515.
[20] R. Benenson, M. Omran, J. Hosang, B. Schiele, Ten years of pedestrian detection, what have we learned? in: ECCV 2014 Workshops, 2014, pp. 613–627.
[21] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3354–3361.
[22] D. Ribeiro, J. Nascimento, A. Bernardino, G. Carneiro, Improving the performance of pedestrian detectors using convolutional learning, Pattern Recognit 61 (2017) 641–649.
[23] S. Wu, R. Laganiere, P. Payeur, Improving pedestrian detection with selective gradient self-similarity feature, Pattern Recognit 48 (8) (2015) 2364–2376.
[24] Z. Zhang, W. Tao, K. Sun, W. Hu, L. Yao, Pedestrian detection aided by fusion of binocular information, Pattern Recognit 60 (2016) 227–238.
[25] C. Lin, C. Chen, W. Hwang, C. Chen, C. Hwang, C. Chang, Novel outline features

for pedestrian detection system with thermal images, Pattern Recognit 48 (11) (2015) 3440–3450.
[26] S. Zhang, R. Benenson, B. Schiele, Citypersons: a diverse dataset for pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4457–4465.
[27] P. Dollar, Z. Tu, P. Perona, S. Belongie, Integral channel features, in: British Machine Vision Conference (BMVC), 2009, pp. 91.1–91.11.
[28] R. Benenson, M. M., Seeking the strongest rigid detector, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 3666–3673.
[29] S. Zhang, C. Bauckhage, A. Cremers, Infromed haar-like features improve pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 947–954.
[30] J. Hosang, M. Omran, R. Benenson, B. Schiele, Taking a deeper look at pedestrians, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 4073–4082.
[31] B. Yang, J. Yan, Z. Lei, S. Li, Convolutional channel features, in: International Conference on Computer Vision (ICCV), 2015, pp. 82–90.
[32] Y. Tian, P. Luo, X. Wang, X. Tang, Pedestrian detection aided by deep learning semantic tasks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 5079–5087.
[33] Z. Cai, M. Saberian, Vasconcelos, A unified multi-scale deep convolutional neural network for fast object detection, in: European Conference on Computer Vision (ECCV), 2016, pp. 354–370.
[34] J. Mao, T. Xiao, Y. Jiang, Z. Cao, What can help pedestrian detection? in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6034–6043.
[35] B. Leibe, E. Seemann, B. Schiele, Pedestrian detection in crowded scenes, in: IEEE Conference on Computer Vision and Pattern Recogntion (CVPR), 2005, pp. 878–885.
[36] X. Wang, T. Han, S. Yan, An hog-lbp human detector with partial occlusion handling, in: International Conference on Computer Vision (ICCV), 2009, pp. 32–39.
[37] W. Ouyang, X. Wang, Single-pedestrian detection aided by multi-pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 3626–3633.
[38] S. Tang, M. Andriluka, B. Schiele, Detection and tracking of occluded people, Int. J. Comput. Vis. 110 (2012) 58–69.
[39] B. Pepik, M. Stark, P. Gehler, B. Schiele, Occlusion patterns for object class detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 3286–3293.
[40] P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, IEEE Trans. Pattern Anal. Mach.Intell. 32 (9) (2010) 1627–1645.
[41] C. Zhou, J. Yuan, Non-rectangular part discovery for object detection, in: British Machine Vision Conference (BMVC), 2014, pp. 22.1–22.13.
[42] B. Wu, R. Nevatia, Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors, in: International Conference on Computer Vision (ICCV), 2005, pp. 90–97.
[43] V. Shet, J. Neumann, V. Ramesh, L. Davis, Bilattice-based logical reasoning for human detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007, pp. 1–8.
[44] K. Divvala, D. Hoiem, J. Hays, A. Efros, M. Hebert, An empirical study of context in object detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 1271–1278.
[45] C. Galleguillos, B. McFee, S. Belongie, G. Lanckriet, Multi-class object localization by combining local contextual interactions, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 113–120.
[46] S. Karaoglu, Y. Liu, T. Gevers, Detect2rank, combining object detectors using learning to rank, IEEE Trans. Image Process. 25 (2016) 233–248.
[47] G. Heitz, D. Koller, Learning spatial context: using stuff to find things, in: European Conference on Computer Vision (ECCV), 2008, pp. 30–43.
[48] C. Desai, D. Ramanan, C. Fowlkes, Discrinative models for multi-class object layout, Int. J. Comput. Vis. 95 (1) (2011) 1–12.
[49] Y. Ding, X. J., Contextual boost for pedestrian detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 2895–2902.
[50] X. Zeng, W. Ouyang, X. Wang, Multi-stage contextual deep learning for pedestrian detection, in: International Conference on Computer Vision (ICCV), 2013, pp. 121–128.
[51] W. Ouyang, X. Zeng, X. Wang, Single-pedestrian detection aided by two-pedestrian detection, IEEE Trans. Pattern Anal. Mach.Intell. 37 (9) (2015) 1875–1889.
[52] J. Yan, X. Zhang, Z. Lei, S. Liao, S. Li, Robust multi-resolution pedestrian detection in traffic scenes, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 3033–3040.
[53] A. Vedaldi, A. Zisserman, Efficient additive kernels via explicit feature maps, IEEE Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 480–492.
[54] C. Toca, M. Ciuc, C. Patrascu, Normalized autobinomial markov channels for pedestrian detection, in: British Machine Vision Conference (BMVC), 2015, pp. 175.1–175.13.
[55] Y. Yang, Z. Wang, F. Wu, Exploring prior knowledge for pedestrian detection, in: British Machine Vision Conference (BMVC), 2015, pp. 176.1–176.12.
[56] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR abs/1409.1556 (2014).

Chunluan Zhou received the B.Eng. degree from Harbin Institute Technology, China, in 2008 and the M.Eng. degree from Zhejiang University, China, in 2011. He is currently pursuing a PhD degree at Nanyang Technoglocigal University. His research interests include object detection, pedestrian detection, computer vision and machine learning.

Junsong Yuan (M'08-SM'14) received the bachelor's degree from the Special Class for the Gifted Young Program of Huazhong University of Science and Technology, Wuhan, China, in 2002, the M.Eng. degree from the National University of Singapore, and the Ph.D. degree from Northwestern University. He is currently an Associate Professor with the School of Electrical and Electronics Engineering, Nanyang Technological University (NTU). His research interests include computer vision, video analytics, gesture and action analysis, and large-scale visual search and mining. He received the Best Paper Award from the International Conference on Advanced Robotics (ICAR17), the 2016 Best Paper Award from the IEEE Transactions on Multimedia, the Doctoral Spotlight Award from the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09), the Nanyang Assistant Professorship from NTU, and the Outstanding EECS Ph.D. Thesis Award from Northwestern University. He is currently a Senior Area Editor of the Journal of Visual Communications and Image Representation and an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. He served as a Guest Editor for the International Journal of Computer Vision. He is the Program Co-Chair of ICME18 and the Area Chair of ACM MM, CVPR, ICIP, ICPR, and ACCV.