Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 767423, 11 pages
http://dx.doi.org/10.1155/2015/767423

Research Article

Random Forest with Adaptive Local Template for Pedestrian Detection

Tao Xiang,[1] Tao Li,[1] Mao Ye,[1] and Zijian Liu[2]

[1]*School of Computer Science and Engineering, Center for Robotics, University of Electronic Science and Technology of China, Chengdu 611731, China*
[2]*Chongqing Jiaotong University, Chongqing 400074, China*

Correspondence should be addressed to Mao Ye; cvlab.uestc@gmail.com

Received 10 May 2015; Revised 27 August 2015; Accepted 11 October 2015

Academic Editor: Panos Liatsis

Pedestrian detection with large intraclass variations is still a challenging task in computer vision. In this paper, we propose a novel pedestrian detection method based on Random Forest. Firstly, we generate a few local templates with different sizes and different locations in positive exemplars. Then, the Random Forest is built whose splitting functions are optimized by maximizing class purity of matching the local templates to the training samples, respectively. To improve the classification accuracy, we adopt a boosting-like algorithm to update the weights of the training samples in a layer-wise fashion. During detection, the trained Random Forest will vote the category when a sliding window is input. Our contributions are the splitting functions based on local template matching with adaptive size and location and iteratively weight updating method. We evaluate the proposed method on 2 well-known challenging datasets: TUD pedestrians and INRIA pedestrians. The experimental results demonstrate that our method achieves state-of-the-art or competitive performance.

1. Introduction

Pedestrian detection is an important instant of object detection. Because of its direct applications in surveillance, intelligent traffic systems, and assisted living [1, 2], it has attracted lots of attention. However, detecting pedestrians with high requirements of real-world applications is still a challenging task due to large intraclass variations caused by different views and articulated poses, partial occlusion, and changes in illumination. In recent years, a number of methods have been proposed to get robust and applied detection. They can be roughly classified into 3 categories, that is, works built on holistic model [3–7], part/patch-based approaches [8–16], and detectors using multiple feature channels and boosted classifier [17–22].

The first category methods take the whole pedestrian as input and make decisions by SVM or template matching. In 2005, Dalal and Triggs [3] proposed Histograms of Oriented Gradients (HOG) feature to encode information of an entire pedestrian, and the detector was trained on HOG features using linear SVM. Since then, some variations [4] and combinations [5, 6] have been proposed to improve the detection performance. In [7, 23], Dominant Orientation Templates (DOT) are used for fast feature calculation, and a holistic detection is defined by template matching. Holistic methods can detect pedestrian fast and accurately in simple scenes; however, the detection performance decreases sharply when the appearance of pedestrian changes due to multiple factors, such as illumination, views, and poses.

The second category methods have two different implementations, that is, Deformable Parts Model (DPM) based [8–11] and Implicit Shape Model (ISM) based [12–16]. DPM [8] and its varieties [9–11] extend the work in [3] with multiple local parts and spatial configurations of these parts by latent SVM. This kind of methods significantly improves detection performance in cluttered scenes. However, the process is time-consuming and some properties of local parts, such as number and size, should be predefined. ISM based methods [12, 13] use small, local image patches to vote for object center with the generalized Hough transform [24]. Hough Forests [14–16] extend the standard Random Forest [25] for learning the codebook of ISM. ISM based methods have been widely

(a) Partial occlusion (b) Multiview (c) Multipose

Figure 1: Examples of pedestrians with different intraclass variations. The red boxes show representative local templates under these conditions.

used for detecting facial feature points [26] and body joints for human pose estimate [27]. However, they get limited success on pedestrian detection because of dense sampling and enormous, scattered votes for the whole human.

The last category methods assemble multiple weak classifiers by boosting algorithm [28]. Each weak classifier is defined by a selected feature channel [17, 18] or a representative exemplar [19]. Particularly, tree structured detectors [20–22] can not only assemble multiple weak classifiers, but also model intraclass subcategories as different branches of the tree. Methods based on tree structured classifiers are good for multiview, multipose pedestrian detection and can obtain very fast detection with cascade architecture [29]. However, the weak classifier cannot divide the sample space optimally because the feature selector and split function used are too simple.

In this paper, we absorb the advantages of approaches mentioned above and try to solve some key problems of pedestrian detection, as shown in Figure 1. It is obvious that all pedestrians are hard to be divided by a holistic property; however, some representative local templates with small varieties can be found. Generally, they are the main parts of a human, such as head, left/right hand, and left/right leg. Motivated by this observation, we propose a new pedestrian detection method which combines multiple weak classifiers built on local templates by means of Random Forest. The templates are adaptively generated with different sizes and different locations in positive exemplars. The splitting functions in the forest are learned by the joint use of template selector and template matching. A weak classifier consists of splitting functions in one depth of the forest. To improve classification accuracy, all weak classifiers are assembled by a boosting-like algorithm [21, 28] with the weights of samples updated iteratively. When a weak classifier is added, the depth of the forest increases until it reaches the predefined maximum depth. For fast calculation, the local template is represented by Dominant Orientation Template (DOT) feature [23]. During detection, a sliding window is passed through each tree, and the final decision is made by averaging estimations of all trees in the forest. To accelerate the detection process, we propose to use cascade detection architecture.

The proposed detection method is evaluated on two well-known pedestrian datasets: TUD pedestrian and INRIA pedestrian, where it achieves state-of-the-art or competitive performance. Our method is on par with the most successful part-based detection system [8]; however, far less design complexity and computation complexity are needed.

The major contributions of our method can be summarized as follows:

(i) We define multiple adaptive local DOT templates with different sizes and different locations to represent the parts of a pedestrian.

(ii) We learn each splitting function in the forest based on template selector and template matching.

(iii) We use a boosting-like algorithm to update the weights of the training samples in a layer-wise fashion.

The rest of this paper is organized as follows. Section 2 describes some related works. Section 3 gives an overview of our method. Section 4 introduces the proposed method. And Section 5 describes the usage of our method, followed by the implementation details and experimental results in Section 6. Section 7 presents our conclusions and future work.

2. Related Work

The most related approaches to ours are the works based on DPM [8–10] which extend the rigid HOG template and SVM approach of [3] with deformable parts and multiple components. In those methods, each deformable part is explicitly defined by a local template and a relative offset vector with respect to the object center. The intraclass variation is captured by dividing the training data into multiple components according to the aspect ratio. The final decision is made by the scores of each template matching minus a deformation cost that depends on the relative position of each part. Nevertheless, the DPM has some disadvantages because

(i) different models have to be trained for each component and (ii) the explicit definitions of local templates and their relative offset vectors are complicated and time-consuming. In contrast, we have a single model that captures the intraclass variability by different branches of the tree. Furthermore, the local parts can be shared between different components, and the position relations between different parts are represented implicitly during assembly.

Some methods [20, 30–32] build on a similar Boosting framework for learning the object models. The influential work on Integral Channel Features [32] computes several feature channels, including color, gradient magnitude, and orientation quantized gradients, which is similar to the DOT feature used in our method. However, the weak classifiers in those methods are only defined by the selected feature channels, and they have not sufficiently made use of the advantages of DPM which has shown state-of-art results on several challenging datasets. In addition, with multiple tree structured classifiers in Random Forest, the weak classifier in our method fully considers multiple splits defined by different local templates, which is more robust to various intraclass variations.

Random Forest has attracted a lot of attention in computer vision. Schulter et al. [21, 22] propose a new Alternating Decision Forest (ADF) classifier for object detection. All trees in the forest are treated as a whole, and the forest is constructed by alternating between training a single depth of the forest and updating the weights of samples for the next depth until the same stopping criterion as in standard Random Forest is reached. Our method adopts a similar way; however, the split function of the forest in ours is defined by local template matching instead of single feature comparison. Yao et al. [33] propose that each node in the forest selects a rectangular region and applies a linear SVM onto the regions of all samples for splitting. Although multiple features are used, the matching of local regions represented by DOT feature can be computed rapidly by bitwise operations. Tang et al. [7] present a new pedestrian detection method combining Random Forest and DOT feature to achieve fast detection; however, the DOT feature is used for representing holistic template which has been proven inflexible for detecting object with intraclass variations.

3. Overview of Our Method

In this section, we give an overview of our method. As shown in Figure 2, it mainly contains extracting DOT feature, generating adaptive local templates and constructing the forest in a layer-wise manner with splitting function defined by template matching.

The first step of our method includes data preparation and DOT feature extraction. The training images denoted as $\mathrm{Im} = \{im_i, l_i\}_{i=1}^{N}$, where im_i is a training sample and l_i is the class label of the sample (−1: negative, 1: positive). For each tree T_k, a training set with N images is sampled in Im by means of bootstrap [34]. Similarly, an exemplar set is randomly sampled from Im^+ (positive samples in Im) with much smaller size than that of Im^+. With these two sample sets prepared for T_k, the corresponding DOT feature set for training set and exemplar set are denoted as $D_k = \{x_i, l_i\}_{i=1}^{N}$ and $Q_k = \{q_m\}_{m=1}^{M}$, respectively. Figure 2(a) illustrates the basic process of extracting DOT feature.

With the data prepared above, the training process begins. Firstly, a few adaptive local templates with different sizes and different locations are generated for each exemplar set Q_k, as shown in Figure 2(b). With the generated local templates, the splitting function at a node in the tree T_k is defined by a selected template and template matching with samples at this node. Given a threshold, the samples can be split into two subsets according to the matching results. The optimal splitting function is found by maximizing the class purity of the divided subsets. Each tree is constructed by splitting the samples recursively until one of the stopping criterions is reached.

To improve classification accuracy, we propose to train the forest in an iterative, layer-wise manner, like boosting algorithm. The iterations are indexed with the current depth of forest $d = 1, \ldots, d_{\max}$, as shown in Figure 2(c). To this end, we define the weight vector of training samples for T_k in depth d as W_k^d, $k = 1, \ldots, K$. It is set uniformly in the first depth and updated iteratively. The class distribution of each node is estimated based on labels and weights of samples. With these definitions, the iterations begin. In iteration d, we firstly find the optimal split functions of nodes in each tree T_k. Then, the samples at each node are split into two child nodes according to the learned splitting functions. Thirdly, a newly weak classifier consisting of the learned splitting functions is added to the trained boosting classifier F_{d-1}. Finally, we use the trained boosting classifier F_d to update the weights of samples in depth $d + 1$ by minimizing a global loss. After all iterations, the training process is finished.

For pedestrian objection, we adopt standard sliding window method in test image. Each window represented by DOT feature is passed through each tree in the trained forest. The final decision is made by averaging estimations of all trees. To accelerate the detection process, we adopt cascade detection architecture to reject negative window as early as possible.

4. Proposed Method

In this section, we firstly introduce some basic concepts and elements of Random Forest [25] since our method is built on Random Forest. Then, we propose a novel splitting function built on adaptive local DOT template. Finally, we give the definition of weak classifier in our method and describe how to assemble multiple weak classifiers in layer-wise fashion by boosting algorithm.

4.1. Introduction to Random Forest. Standard Random Forest [25] is an ensemble of K randomized binary decision trees $\{T_k(x)\}_{k=1}^{K} : X \to Y$, which describe a nonlinear mapping from Z-dimensional feature space $X = R^Z$ to label space $Y = \{-1, 1\}$ (although the Random Forest is inherently multiclass, we only consider the binary for pedestrian detection). Given a sample $x \in X$, each tree returns a score defined by class

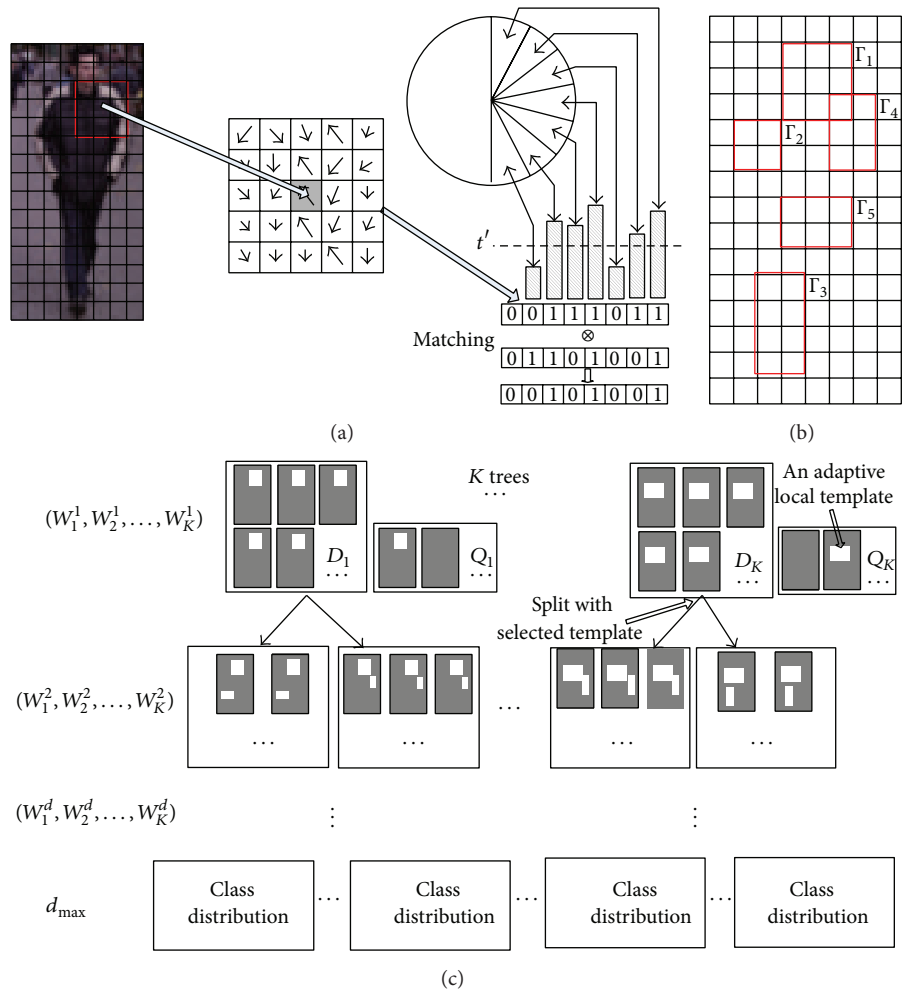


Figure 2: (a) Illustration of DOT feature extraction and matching, (b) proposed template selector, and (c) training the forest in a layer-wise manner.

probability distribution $p_k(y \mid x)$; the final class label $y^* \in Y$ is obtained via maximizing the total average score of K trees:

$$y^* = \arg\max \frac{1}{K} \sum_{k=1}^{K} p_k (y \mid x). \quad (1)$$

Random Forest assembles multiple trees by means of bootstrap [34]. The trees in forest are constructed independently from each other by recursively splitting samples at each node such that the class-label purity of samples reaching the newly created child node increases, until one of the following stopping criterions is met: (1) the depth of node is equal to the maximal one; (2) the number of samples reaching the node is too small; (3) the class-label purity of samples reaching the node is high enough.

Generally, a splitting function is parameterized by two values, a selected feature dimension φ, and a threshold τ. The splitting function is then written as

$$s(x; \varphi, \tau) = \begin{cases} 0 & \text{if } \varphi(x) > \tau \\ 1 & \text{otherwise,} \end{cases} \quad (2)$$

where $s(x; \varphi, \tau)$ defines which child node the sample x reaches.

Each node chose the best splitting function $\Omega^* = (\varphi^*, \tau^*)$ out of random sampled set by maximizing class-label purity defined by

$$I(\Omega) = \frac{|S^L|}{|S^L| + |S^R|} \cdot H(S^L) + \frac{|S^R|}{|S^L| + |S^R|} \cdot H(S^R), \quad (3)$$

where S^L and S^R are the sets of samples which reach the left and right child node, respectively, according to $s(x; \Omega)$. $|\cdot|$ denotes the size of a set, and $H(\cdot)$ measures the class-label purity of a sample set. In this paper, we use negative entropy to calculate $H(\cdot)$, which is defined as

$$H(S) = \sum_{c \in \{-1,1\}} p(c \mid S) \log(p(c \mid S)). \quad (4)$$

Here, $p(c \mid S)$ is the probability for class c, estimated by the ratio of positive or negative samples in S.

4.2. A Novel Split Function with Adaptive Local DOT Template. The key point of Random Forest based detector is to design

a fast and effective splitting function. As discussed in [25], a nonlinear hyperplane outperforms axis-aligned ones. In this section, we propose a novel splitting function which is defined by adaptive local DOT template and nonlinear template matching.

The prerequisite of the proposed splitting function is to compute DOT feature. As described in [35], we firstly give a brief introduction about DOT feature extraction in this paper. DOT feature is a block based descriptor, and each 7×7 pixels block encodes the discretized gradient orientations of the 7 strongest gradients into one byte. With a defined threshold, the first bit indicates whether this block is uniform, and the 7 dominant orientations are quantized into the remaining 7 bits. In order to make the matching invariant to small local deformations, $2D$ translations in the range $[-3, +3]$ for each block are considered. Figure 2(a) gives an illustration of computing DOT feature. In order to tolerate changes caused by colors and illuminations, similar to [7], we also encode the HSV color space in the similar way; that is, each block encodes the discretized H value of the 7 strongest V into one byte. The final binary representation of each block is 16-bit descriptor formed by concatenating dominant orientations descriptor and dominant colors descriptor. For simplification, we call the representation built on two mentioned DOT-based descriptors DOT feature. The similarity of DOT template and training image described by DOT feature is defined by bitwise AND operations.

We suppose that all training images can be partitioned into $O \times P$ overlapping blocks, and each block is encoded as 16-bit DOT feature. $S = \{x_i, l_i\}_{i=1}^{N_s}$ is the DOT training set at a node in tree T_k. $Q_k = \{q_m\}_{m=1}^{M}$ is the DOT exemplar set of T_k, and each q_m is a holistic DOT template since it describes a whole pedestrian. The adaptive local DOT template is generated using a local template selector illustrated in Figure 2(b). The basic idea is as follows. Firstly, an exemplar q_m is randomly selected in Q_k. Then, an adaptive local DOT template Γ is generated by randomly selecting a rectangular region including $B \times C$ contiguous blocks in q_m. The top-left coordinate of Γ is denoted as (u, v), and the width $B \in [1, L]$ and height $C \in [1, L]$ are randomly generated with the predefined maximum size L. Note that the coordinate here is based on blocks.

With the selected adaptive local template Γ and configuration $\Pi = (u, v, B, C)$, S is divided by comparing local template Γ with local DOT features of all training samples, and each local DOT feature is computed according to configuration Π. Therefore, the splitting function in (2) becomes

$$s(x; \Gamma, \Pi, \tau) = \begin{cases} 0 & \text{if } \chi(\Gamma, x(\Pi)) > \tau \\ 1 & \text{otherwise,} \end{cases} \tag{5}$$

$$\chi(\Gamma, x(\Pi)) = \sum_{b=1}^{B} \sum_{c=1}^{C} \Delta\left(\Gamma_{b,c} \otimes x(\Pi)_{b,c}\right).$$

Here, $x(\Pi)$ is the local DOT feature in training sample x with configuration Π; χ is the matching function of DOT feature; $\otimes$ is the bitwise AND operation; $\Gamma_{b,c}$ and $x(\Pi)_{b,c}$ are 16-bit DOT feature for a block at location $[b, c]$ in Γ and $x(\Pi)$, respectively; Δ is used for counting the number of 1 in 16-bit matching result. That is to say, the similarity is measured by the number of matched dominant orientations and dominant colors in Γ and $x(\Pi)$. The optimal split is parameterized by $\Omega^* = (u^*, v^*, B^*, C^*, \Gamma^*, \tau^*)$, which is optimized by maximizing class-label purity of each division. Algorithm 1 gives an overview of the optimization process.

With the proposed adaptive local DOT template, the splitting function is not only robust to small local transformations, but also very fast to evaluate since the matching function can be further sped up using SSE operations, similar to [35]. More importantly, each tree in the forest provides both discriminative and complementary local information for classification.

4.3. Assembling Weak Classifiers with a Global Loss. The tree structured methods typically take the splitting function in each depth as a weak classifier, and multiple weak classifiers are assembled with boosting algorithm [19–21, 28]. In order to make full use of the complementary information provided by multiple trees, we generalize this idea to Random Forest based method. To this end, the forest is treated as a whole and constructed in layer-wise fashion. Each layer is indexed with current depth of the forest d. The split functions in one depth of the forest constitute a weak classifier. For assembling multiple weak classifiers with boosting method, we introduce a global loss. Suppose that the boosting classifier F_d consisting of the first d weak classifiers has been trained. It gives a predication about the class distribution of each sample. The new $(d+1)$th weak classifier f_{d+1} is learned and added by minimizing the global loss computed by F_d. With a weak classifier added, the forest grows until it reaches the maximum depth.

As described in Section 3, the forest includes K trees with maximum depth $d_{\max}$. Each tree T_k has a training set $D_k = \{x_i, l_i\}_{i=1}^{N}$ and an exemplar set $Q_k = \{q_m\}_{m=1}^{M}$. To obtain the final boosting classifier, the training procedure of boosting runs $d_{\max}$ times. Different from standard Random Forest, the samples in our method are weighted and their weights are updated in each depth. The initial weight vector of each D_k is set uniformly in $d = 1$, denoted as $W_k^1 = [w_{k,1}^1, w_{k,2}^1, \ldots, w_{k,N}^1]$, and $w_{k,i}^d$ is the weight of the ith sample in D_k with current depth d. With weighted samples, the class distribution of a node used in (4) should take the weights into account. It is defined as

$$p\left(c \mid S'\right) = \frac{\sum_{r=1}^{|S'|} \delta\left[l_r = c\right] \cdot w_r}{\sum_{r=1}^{|S|} w_r}, \tag{6}$$

where S' is a sample set, w_r is the weight of rth sample in S', and $\delta[l_r = c]$ is an indicator function which returns 1 if $l_r = c$ and 0 otherwise. Then the class distribution of each node in the first layer can be computed by (6).

With the initial weights and class distributions, the splitting function of each node in depth $d = 1$ can be learned by Algorithm 1. Suppose the forest with current

Input:
Samples at a node in T_k: $S = \{x_i, l_i\}_{i=1}^{N_s}$
Exemplar set of T_k: $Q_k = \{q_m\}_{m=1}^{M}$
Block size of each sample: $O \times P$
Maximum size of local template: L
Output:
Optimal splitting parameter: $\Omega^* = (u^*, v^*, B^*, C^*, \Gamma^*, \tau^*)$
(1) Initialization: $u^*, v^*, B^*, C^*, \Gamma^*, \tau^*, I^* = -\infty$
(2) **for** each $q_m \in Q_k$ **do**
(3) **for** $iter1 = 1$ to max$Iter1$ **do**
(4) Randomly generate configuration:
$\Pi = (u, v, B, C), u \in [1, O], v \in [1, P], B, C \in [1, L]$
(5) Generate adaptive local DOT template Γ in q_m according to Π
(6) Calculate the maximum and minimum value of τ:
$\tau_{\max} = \max\{\chi(\Gamma, x_i(\Pi)), x_i \in S\}$
$\tau_{\min} = \min\{\chi(\Gamma, x_i(\Pi)), x_i \in S\}$
(7) **for** $iter2 = 1$ to max$Iter2$ **do**
(8) Randomly select a threshold $\tau \in [\tau_{\min}, \tau_{\max}]$
(9) Divide samples at S into two subset:
$S^L = \{x_i \in S : \chi(\Gamma, x_i(\Pi) \geq \tau\}$
$S^R = \{x_i \in S : \chi(\Gamma, x_i(\Pi) < \tau\}$
(10) Calculate class-label purity I by (3)
(11) **if** $I > I^*$
(12) $I^* = I$
(13) $u^* = u, v^* = v, B^* = B, C^* = C, \Gamma^* = \Gamma, \tau^* = \tau$
(14) **end if**
(15) **end for**
(16) **end for**
(17) **end for**

Algorithm 1: Learning splitting function with local template selector.

depth $d \in [1, d_{\max}]$ has been trained. It can be considered as a boosting classifier, written as

$$F_d(x_i; \Theta_d) = \sum_{t=1}^{d} \beta \cdot f_t(x_i; \theta_t). \tag{7}$$

Here, $f_t(x_i; \theta_t)$ is the weak classifier in depth t; the trained boosting classifier and each weak classifier are parameterized by Θ_d and θ_t, respectively; β is shrinkage factor [28]. F_d can be estimated by the trained forest with depth d:

$$F_d(x_i; \Theta_d) = \sum_{k=1}^{K} \frac{1}{K} p(c = 1 \mid S_{k,d}), \tag{8}$$

where $S_{k,d}$ is a sample set at a node where x_i is routed by tree T_k in depth d. If $d < d_{\max}$, $f_{d+1}(x_i; \theta_{d+1})$ will be trained and added to F_d in depth $d + 1$. The assembled strong boosting classifier becomes

$$F_{d+1}(x_i; \Theta_{d+1}) = F_d(x_i; \Theta_d) + f_{d+1}(x_i; \theta_{d+1}). \tag{9}$$

As discussed in [21, 28], training the new weak classifier can be written as global loss minimization problem:

$$\arg\min_{\theta_{d+1}} \sum_{i=1}^{N} \text{loss}(l_i; F_d(x_i; \Theta_d) + f_{d+1}(x_i; \theta_{d+1})), \tag{10}$$

where loss$(\cdot)$ is a differentiable loss function; Θ_d is parameter set fixed already; and θ_{d+1} is parameter set to be trained in depth $d + 1$. The minimization problem can be solved by updating the weights of samples in each tree with depth $d + 1$:

$$w_{k,i}^{d+1} = \left| \frac{\partial \text{loss}(l_i, F_d(x_i; \Theta_d))}{\partial F(x_i)} \right|. \tag{11}$$

With the updated weights of samples in depth $d + 1$, θ_{d+1} including parameters of each split function in f_{d+1} can be learned by Algorithm 1.

In this paper, the nonconvex tangent loss function proposed by Masnadi-Shirazi et al. [23] is adopted. It is defined as

$$\begin{aligned} \text{loss}(l_i, F_d(x_i; \Theta_d)) &= (2 \arctan(l_i \cdot \rho(x_i)) - 1)^2, \\ \rho(x_i) &= \tan(F_d(x_i; \Theta_d) - 0.5). \end{aligned} \tag{12}$$

Although any differentiable loss function can be used, the tangent loss function is proven more robust to label noise. With the tangent loss function defined above, (11) becomes

$$w_{k,i}^{d+1} \propto \begin{cases} 8(1 - F_d(x_i; \Theta_d)) & \text{if } l_i = 1 \\ 8F_d(x_i; \Theta_d) & \text{if } l_i = -1. \end{cases} \tag{13}$$

Input:
Number of trees: K
Maximum depth: $d_{\max}$
Training set for each tree: $D_k = \{x_i, l_i\}_{i=1}^{N}$
Template set for each tree: $Q_k = \{q_m\}_{m=1}^{M}$
(1) Initialize weights for samples of each tree:
$W_k^1 = (w_{k,1}^1 = 1/N, \ldots, w_{k,N}^1 = 1/N)$
(2) Compute class distribution of root node of each tree by (6)
(3) **for** $d = 1$ to $d_{\max}$ **do**
(4) Check stopping criterions for nodes in depth d
(5) Split nodes in depth d by Algorithm
(6) Update weight $w_{k,i}^d$ by (13) for each sample in each D_k
(7) Calculate class distribution of newly created nodes in depth $d + 1$ by (6)
(8) **end for**

Algorithm 2: Procedure of assembling weak classifiers.

When the weak classifier in $d_{\max}$ has been trained, the construction of the forest stops. The training procedure is summarized in Algorithm 2.

5. Detecting Pedestrian with Proposed Method

In order to detect objects, we adopt a standard sliding window method in test image represented by DOT feature. Given a test window ω, it is passed through each tree T_k in the trained forest according to the learned split parameters of each node, until reaching a leaf node Leaf_k in T_k. The score of window ω estimated by T_k is computed by class distribution of Leaf_k, which is calculated by (6) with $c = 1$. The final score of window ω estimated by the forest is defined by averaging all scores obtained by trees in the forest:

$$\text{Score}(\omega) = \frac{1}{K}\sum_{k=1}^{K} T_k(\omega), \tag{14}$$

$$T_k(\omega) = \pi\left(c = 1 \mid \text{Leaf}_k\right). \tag{15}$$

The test window ω is classified as a pedestrian if $\text{Score}(\omega)$ exceeds the detection threshold ξ which is found during the validation.

Particularly, we adopt cascade architecture to speed up the detection procedure. Using this approach, the windows which theoretically cannot achieve threshold ξ are rejected as early as possible. The cascade architecture provides a significantly fast detection due to the fact that there is no need to compute the probability for all trees in the forest for a large majority of windows in the test image. Algorithm 3 describes the cascade detection procedure.

Input:
Train forest: $\{T_k(x)\}_{k=1}^{K}$
Test window: ω
Output:
Label of ω
(1) Initialize: score = 0
(2) **for** $k = 1$ to K **do**
(3) Evaluate ω with T_k by (15)
(4) score = score + $T_k(\omega)$
(5) $\text{score}^* = \sum_{i=k+1}^{K} 1.0 = K - k$
(6) **if** score + $\text{score}^* < K \cdot \xi$ **then**
(7) reject ω
(8) **return** −1
(9) **end if**
(10) **end for**
(11) **return** 1

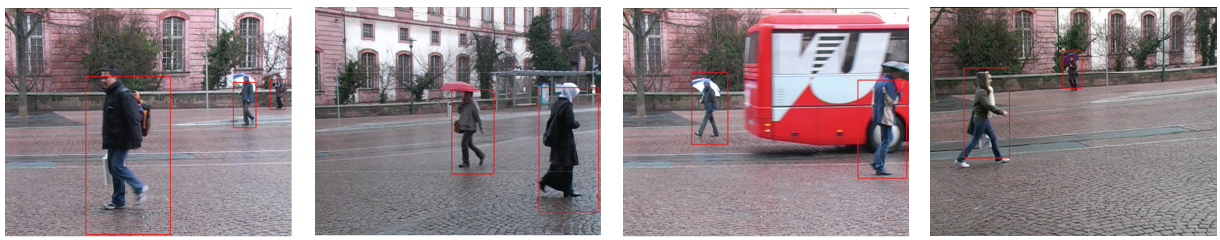
Algorithm 3: Cascade detection procedure.

6. Experimental Results

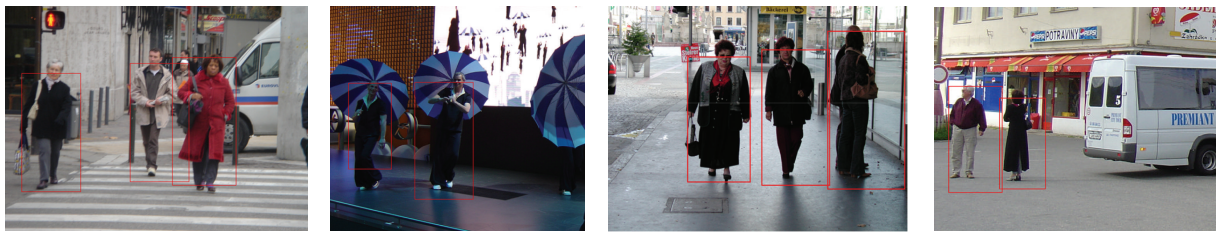
We evaluate the proposed method on two challenging pedestrian datasets: TUD pedestrians, INRIA pedestrians, where we provide a performance comparison with the other competing detection methods, including the best algorithms (as far as we know) in this field. We follow the PASCAL protocol [36] to decide whether the detected object is true positive; namely, the overlap area of detected bounding box and the ground-truth exceeds 50%. In order to avoid multiple detections for the same ground-truth, we reject the detections with centers inside the bounding boxes detected with higher score. Additionally, we analyze the two most relevant parameters of our method on TUD pedestrian dataset: the maximum size of local template L and the maximum depth $d_{\max}$. All the experiments are performed with two Inter Core(TM) i5 3.2 GHz CPU, 16 G RAM, and Windows 64-bit OS.

6.1. Datasets and Experimental Setup

6.1.1. TUD Pedestrians. The TUD pedestrian dataset is a widely used benchmark for human detection. This dataset includes 400 training images and 250 test images with 311 pedestrians. Because the background in this dataset is mainly street; moreover, the diversities of backgrounds are low; we suggest collecting negative samples from INAIA dataset. In



(a) Detection examples on TUD pedestrians dataset



(b) Detection examples on INRIA pedestrians dataset

Figure 3: Detection examples on TUD pedestrians and INRIA pedestrians.

addition, we randomly select some positive samples from INAIA dataset.

6.1.2. INRIA Pedestrians. The INRIA pedestrian dataset is also a popular benchmark for pedestrian detection. This dataset is very challenging because of various intraclass variations and cluttered scenes. The training set includes 614 images with 1208 pedestrians and 1218 background images. In order to tolerate changes caused by poses, views, occlusions, and so forth, we flip the 1208 normalized pedestrian windows and get 2416 normalized positive samples. Negative training windows are sampled randomly from 1218 background images. The test set includes 288 images with 1126 pedestrians and 453 images without them.

During training the proposed model for pedestrian detection on these two datasets, all samples are normalized to 51×95 pixels. As mentioned before, the size of each block is 7×7 pixels. The overlap of the neighboring blocks is 4 pixels. Therefore, each sample can be partitioned into 12×23 overlapping blocks. We set the number of trees $K = 100$, as discussed in [22]. Regarding the maximum depth $d_{\max}$ and the maximum size of local template L, we set $d_{\max} = 8$ and $L = 6$ which are exhaustively optimized using a validation set. Additionally, we find that the discriminating power of local template with small size is too low. We reset the range of width and height of each local template as $B, C \in [3, L]$. During detection, the test image is partitioned in the same way, and each detection window slides with a block. To handle scale variations of object, we resized a test image to 20 scales with stride 1.05.

6.2. Experimental Results. Figures 3(a) and 3(b) demonstrate some detection examples of our method on TUD pedestrian and INRIA pedestrian dataset, respectively. They strongly prove that the proposed algorithm can detect people with large intraclass variability caused by different poses, size, and clothing under varying illumination and cluttered scenes.

To evaluate the performance of different methods evaluated on TUD pedestrian test set, the Receiver Operating Characteristic (ROC) curves are drawn to describe the statistical comparison of different methods. We use the definition in [36] that Recall and Precision are computed as

$$\begin{aligned} \text{Recall} &= \frac{\text{Tp}}{\text{nP}}, \\ \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \end{aligned} \tag{16}$$

where TP and FP are the number of true positive and false positive, respectively, during test and nP is the total number of positive in the test dataset. The goal of all detection methods is to improve the Recall and, in the meanwhile, also to improve the Precision. Unfortunately, they are mutually related to each other and mutually restrict each other. Generally, Area Under Curve (AUC) is used for measuring the performance of different methods according to corresponding ROC curves. The comparison result is shown in Figure 4. Applying the proposed method to detect pedestrian on this dataset achieves AUC = 0.908, which outperforms other competing algorithms. Furthermore, we compute the Equal Error Rate (EER) of different methods, as shown in Table 1. EER is the point on the ROC curve that corresponds to have an equal probability of missclassifying a positive or negative sample.

The INRIA pedestrian test set contains pedestrians with large intraclass variability. The statistical comparison of different methods is defined by miss rate at 1 false positive per image (FPPI). We follow the definition in [3] that the miss rate is computed as

$$\text{miss rate} = \frac{\text{NF}}{\text{nP}}, \tag{17}$$

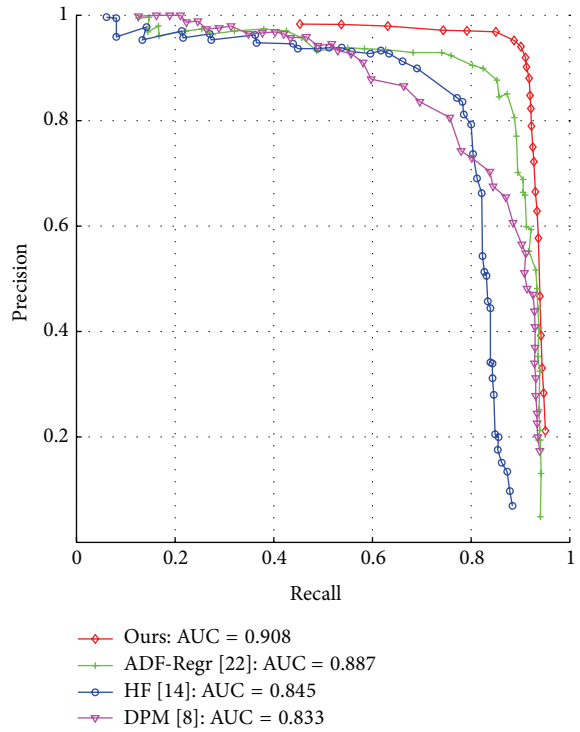


Figure 4: Performance curves of different methods evaluated on the TUD pedestrian dataset.

Table 1: EER of different methods evaluated on TUD pedestrian dataset.

Algorithm	Equal Error Rate (EER)
Ours	0.911
ADF-Regr [22]	0.879
HF [14]	0.796
DPM [8]	0.783

where FN is the false negative during test. For pedestrian detection, if the threshold of the classifier is low, the miss rate will decrease; at the same time, the number of false positive in each test image will increase. To make fair comparison, we should specify a statistical indicator and use another indicator to evaluate the performance of different methods. Generally, miss rate at FPPI $=$ 1 is used, as shown in Table 2. The proposed method achieves miss rate of 0.12 at 1 FPPI, which is not as good as that of the state-of-the-art method [18]. Yet it is still quite competitive and, in particular, performs better than methods built on other block based descriptors, such as HOG and LBP. That gives direct evidence to the effectiveness of the proposed adaptive local DOT template. Regarding the gap between proposed method and the work in [18], the main reason is that the detector in [18] is built not only on local features, but also on the full object.

Regarding detection speed, we evaluate it on TUD pedestrian dataset. With fast DOT template matching and cascade detection architecture in our method, the mean detection time of one test image achieves 0.18 second which is faster than the HF's 1.11 second and DPM's 0.85 second.

Table 2: Performances of different methods evaluated on INRIA pedestrian dataset at 1 FPPI.

Algorithm	Miss rate at 1 FPPI
Ours	0.12
Parts + dictionary [11]	0.12
Very fast [18]	0.07
HOG-LBP [6]	0.14
HOG [3]	0.23

6.3. Parameter Evaluation. In this section, we analyze the two most important parameters of the proposed method: the maximum size of local template L and the maximum depth $d_{\max}$. The TUD pedestrian dataset is used for evaluation. To evaluate the maximum size of local template L, we fix the maximum depth $d_{\max} = 8$. Generally, L represents the compromise between discrimination and robustness to local variations. If L is too small, the discriminating power of selected local template is low; while L is too large, local DOT template cannot tolerate variations caused by different views and articulated poses, partial occlusion, and changes in illumination. We depict the relations between L and performance measured by AUC of PR curve in Figure 5(a). As expected, the performance increases with L up to a certain limit $L = 6$ and decreases if $L > 6$. To evaluate the second parameter $d_{\max}$, we fix the maximum size of local template $L = 6$. From another point of view, $d_{\max}$ can be considered as the number of local templates in pedestrian assembled for classification. It is affected by two factors: the representativeness of selected templates and the way of assembling multiple weak classifiers built on these templates. The experimental results show that $d_{\max} = 8$ is enough for our method, as shown in Figure 5(b).

7. Conclusion and Future Work

We have proposed a novel compositional model for pedestrian detection in cluttered scenes. The key idea of our method is to assemble multiple weak classifiers which are defined by adaptive local templates. We achieve it by Random Forest. The forest is built in an iterative, layer-wise manner. The adaptive local templates are used for learning splitting functions in the forest, and all splitting functions in one depth form a weak classifier. Each newly weak classifier is learned and added by minimizing a global loss, with weights of samples updating. The final experimental results on two challenging pedestrian datasets indicate that the proposed method achieves the state-of-the-art or competitive performance.

As demonstrated in this paper, the extensive experiments show that our method is robust and effective for detecting pedestrians with various intraclass variations to some extent. However, we have to concede that there is room for improvement, particularly on challenging INRIA pedestrian dataset.

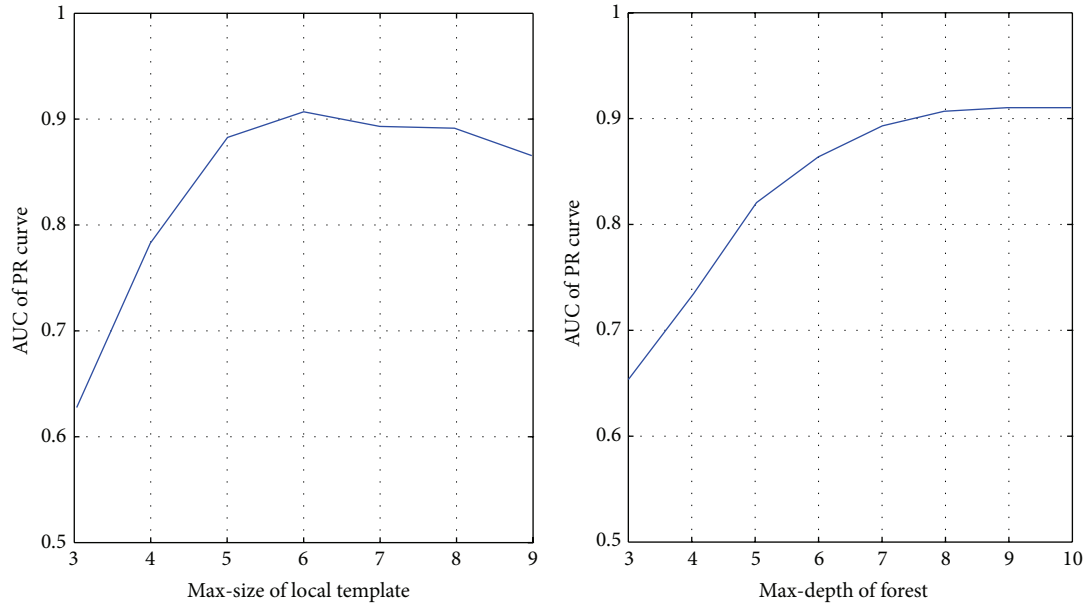


Figure 5: (a) Evaluation of the maximum size of local template. (b) Evaluation of the maximum depth of forest. We plot the performance as AUC of PR curve. The TUD pedestrian dataset is used.

The key of the problem is to model the combination relations of selected local templates explicitly during learning each weak classifier, which is used for providing information about poses, views, and occlusions. In the future works, we will continue our researches to solve this problem.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (61375038 and 11401060).

References

[1] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: an evaluation of the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2012.

[2] D. Gerónimo, A. M. López, A. D. Sappa, and T. Graf, "Survey of pedestrian detection for advanced driver assistance systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1239–1258, 2010.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 886–893, IEEE, San Diego, Calif, USA, June 2005.

[4] W. R. Schwartz, A. Kembhavi, D. Harwood, and L. S. Davis, "Human detection using partial least squares analysis," in *Proceedings of the IEEE 12th International Conference on Computer Vision*, pp. 24–31, Kyoto, Japan, September 2009.

[5] S. Walk, N. Majer, K. Schindler, and B. Schiele, "New features and insights for pedestrian detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 1030–1037, San Francisco, Calif, USA, June 2010.

[6] X. Wang, T. X. Han, and S. Yan, "An HOG-LBP human detector with partial occlusion handling," in *Proceedings of the 12th IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 32–39, IEEE, Kyoto, Japan, September-October 2009.

[7] D. Tang, Y. Liu, and T. K. Kim, "Fast pedestrian detection by cascade random forest with dominant orientation template," in *Proceedings of the British Machine Vision Conference (BMVC '12)*, pp. 1–11, Surrey, Canada, September 2012.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[9] Y. Yang and D. Ramanan, "Articulated human detection with flexible mixtures-of-parts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2878–2890, 2013.

[10] H. Pirsiavash and D. Ramanan, "Steerable part models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 3226–3233, IEEE, Providence, RI, USA, June 2012.

[11] C. Yao, X. Bai, W. Liu, and L. J. Latecki, "Human detection using learned part alphabet and pose dictionary," in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V*, vol. 8693 of *Lecture Notes in Computer Science*, pp. 251–266, Springer, Berlin, Germany, 2014.

[12] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International Journal of Computer Vision*, vol. 77, no. 1–3, pp. 259–289, 2008.

[13] A. Lehmann, B. Leibe, and L. Van Gool, "Fast PRISM: branch and bound hough transform for object class detection," *International Journal of Computer Vision*, vol. 94, no. 2, pp. 175–197, 2011.

[14] J. Gall and V. Lempitsky, "Class-specific hough forests for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '09)*, pp. 1022–1029, IEEE, Miami, Fla, USA, June 2009.

[15] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky, "Hough forests for object detection, tracking, and action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2188–2202, 2011.

[16] P. Wohlhart, S. Schulter, M. Köstinger, P. M. Roth, and H. Bischof, "Discriminative Hough forests for object detection," in *Proceedings of the 23rd British Machine Vision Conference (BMVC '12)*, pp. 23–35, September 2012.

[17] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[18] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, "Pedestrian detection at 100 frames per second," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 2903–2910, IEEE, Providence, RI, USA, June 2012.

[19] H. Li, Z. Lin, J. Brandt, X. Shen, and G. Hua, "Efficient boosted exemplar-based face detection," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 1843–1850, Columbus, Ohio, USA, June 2014.

[20] B. Wu and R. Nevatia, "Cluster boosted tree classifier for multi-view, multi-pose object detection," in *Proceedings of the IEEE 11th International Conference on Computer Vision (ICCV '07)*, pp. 1–8, Rio de Janeiro, Brazil, October 2007.

[21] S. Schulter, C. Leistner, P. Wohlhart, P. M. Roth, and H. Bischof, "Alternating regression forests for object detection and pose estimation," in *Proceedings of the 14th IEEE International Conference on Computer Vision (ICCV '13)*, pp. 417–424, IEEE, Sydney, Australia, December 2013.

[22] S. Schulter, C. Leistner, P. Wohlhart, P. M. Roth, and H. Bischof, "Accurate object detection with joint classification-regression random forests," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 923–930, IEEE, Columbus, Ohio, USA, June 2014.

[23] H. Masnadi-Shirazi, V. Mahadevan, and N. Vasconcelos, "On the design of robust classifiers for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 779–786, IEEE, San Francisco, Calif, USA, June 2010.

[24] D. H. Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[25] A. Criminisi and J. Shotton, *Decision Forests for Computer Vision and Medical Image Analysis*, Springer, London, UK, 2013.

[26] M. Dantone, J. Gall, G. Fanelli, and L. Van Gool, "Real-time facial feature detection using conditional regression forests," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 2578–2585, IEEE, Providence, RI, USA, June 2012.

[27] M. Sun, P. Kohli, and J. Shotton, "Conditional regression forests for human pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 3394–3401, IEEE, Providence, RI, USA, June 2012.

[28] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[29] L. Bourdev and J. Brandt, "Robust object detection via soft cascade," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 2, pp. 236–243, June 2005.

[30] R. Benenson, M. Mathias, T. Tuytelaars, and L. Van Gool, "Seeking the strongest rigid detector," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, pp. 3666–3673, Portland, Ore, USA, June 2013.

[31] P. Dollar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.

[32] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in *Proceedings of the British Machine Vision Conference (BMVC '09)*, pp. 56–68, London, UK, September 2009.

[33] B. Yao, A. Khosla, and L. Fei-Fei, "Combining randomization and discrimination for fine-grained image categorization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 1577–1584, Providence, RI, USA, June 2011.

[34] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[35] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab, "Dominant orientation templates for real-time detection of texture-less objects," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 2257–2264, IEEE, San Francisco, Calif, USA, June 2010.

[36] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.