Cluster Boosted Tree Classifier for Multi-View, Multi-Pose Object Detection

Bo Wu and Ram Nevatia
University of Southern California
Institute for Robotics and Intelligent Systems
Los Angeles, CA 90089-0273
$\{bowu|nevatia\}@usc.edu$

Abstract

Detection of object of a known class is a fundamental problem of computer vision. The appearance of objects can change greatly due to illumination, view point, and articulation. For object classes with large intra-class variation, some divide-and-conquer strategy is necessary. Tree structured classifier models have been used for multi-view multi-pose object detection in previous work. This paper proposes a boosting based learning method, called Cluster Boosted Tree (CBT)*, to automatically construct tree structured object detectors. Instead of using predefined intra-class sub-categorization based on domain knowledge, we divide the sample space by unsupervised clustering based on discriminative image features selected by boosting algorithm. The sub-categorization information of the leaf nodes is sent back to refine their ancestors' classification functions. We compare our approach with previous related methods on several public data sets. The results show that our approach outperforms the state-of-the-art methods.*

1. Introduction

Detection of objects of a known class is a fundamental problem of computer vision. The image appearance of objects changes due to many factors, such as illumination, view points, and poses. To make the image patterns relatively invariant, sometimes the external variations are limited. For example, the class of frontal human faces has smaller variance than the class of multi-view faces. For object classes with small intra-class variation, the cascade structured classifier proposed by Viola and Jones [18] has proven to be an efficient solution. However, for more diverse patterns, such as multi-view faces, or multi-view multi-pose human bodies, more powerful classifier models are needed, such as the tree structured classifiers in [10, 11]. The underlying philosophy is "divide-and-conquer": when the class can not be modeled as a whole, divide it into several sub-categories, and learn a model for each of them individually.

When the appearance changes due to multiple factors, as in the case of human bodies, it is hard to find one dominating property to divide the samples. Manually assigning the sub-category labels for the training samples could be difficult and time consuming. Also the domain knowledge based sub-categorization may not be optimal for the classification task. In this paper, we propose a learning method that automatically constructs tree structured classifiers. Instead of using predefined intra-class sub-categorization based on domain knowledge, we divide the sample space by unsupervised clustering based on discriminative image features. We call this method *Cluster Boosted Tree* (CBT).

1.1. Related work

The problem of object detection has been worked on since the beginning of computer vision research. A recent breakthrough was the cascade structured detector proposed by Viola and Jones [18]. When the intra-class variation is small, *e.g*. the frontal view faces in [18] and the frontal/rear view pedestrians in [12], the cascade structured detectors achieve good accuracy. However, when the object class variance is high, a divide-and-conquer strategy is necessary, *e.g*. the variations of tree structured detectors for multi-view faces [11, 17].

The differences between these tree based methods are mainly two fold: first, how to split a tree node; and second, how to learn a tree node. In [17], a pose estimator is learned to send each sample into one of several view based sub-categories; while in [11] one tree node is a multi-class multi-label classifier, where each sub-class corresponds to one view and a sample may be sent to multiple view-based sub-classes depending on the output vector. The tree nodes are learned by a binary AdaBoost [21] based learning framework in [17, 10], and by Vector Boosting in [11] (we refer to the method of [11] as *Vector Boosted Tree*, VBT). Among these, the method of [11] achieved the best results. We based our approach on this classifier model.

Intra-class sub-categorization can be either top-down or bottom up. For top-down, we make use of some domain knowledge to group the samples. For example, Huang *et al*. [11] pre-define five view categories based on the left-right out-of-plane rotation angle of the faces. The top-down

978-1-4244-1631-8/07/$25.00 ©2007 IEEE

sub-categorization is usually done manually, as the automatic extraction of high level knowledge could be an even harder problem. Some methods have been developed to do the intra-class sub-categorization automatically based on low level image features, *i.e*. bottom-up. Seemann *et al*. [3] cluster pedestrians based on their silhouettes. However, the features used for clustering are different from the features used for detection, hence the clustering is not optimized directly for the detection task. Tu [10] proposes a *Probabilistic Boosting-Tree* (PBT) model for general classification problems, in which the estimated posterior probability is used to divide the samples. Our experiments show that splitting based on classification confidence can not achieve balanced division for both the positive and negative sample spaces, for the asymmetric classification problem of object detection.

Some other researchers try to use examplar-based methods to model complicated object classes. Shan *et al*. [4] learn a pedestrian detector by boosting examplar-based weak classifiers. Each weak classifier is built based on one representative examplar. However their algorithm is sensitive to the initial candidate set of the examplars. How to select the candidate set is still an open problem. SVM based approaches, *e.g*. the Histogram of Oriented Gradients (HOG) descriptor based SVM for pedestrian detection by Dalal and Triggs [9], is another type of examplar based method. Although SVM based methods have good accuracy, they are computationally expensive.

1.2. Outline of our approach

For our classifier model, we choose the tree structure proposed by Huang *et al*. [11]. The VBT model is more suitable for detection tasks than PBT, as PBT does not have the cascade detection strategy, and the root node of PBT is ignorant of the intra-class sub-categorization information.

The learning method in [11] needs the user to predefine the intra-class sub-categories and the tree structure, label the training samples accordingly, and choose the position to split. These require a lot of experience and are not easy to generalize from one application to another. We propose a Cluster Boosted Tree method to automatically construct boosted tree classifiers.

Our learning method is an iterative algorithm. At the beginning, the tree classifier contains only a null branch, *i.e*. there is one sub-category containing all the samples. At each boosting round, for each branch of the tree, one simple feature based weak classifier is selected from the weak hypothesis space to attached to the branch. Then the discriminative power of the newly added weak classifier is estimated. If this power is too weak, we divide the branch into two smaller and thus easier problems by unsupervised clustering, *i.e*. the training samples are divided into two subcategories. The unsupervised clustering is done based on the responses of the selected image features, which form an informative descriptor of the samples. The new intra-class sub-categorization information is used to generate new tree branches, and also propagated upstream to refine the classification functions of the parent nodes. The growth of each branch stops automatically when a target accuracy is reached. We apply our method to the classes of pedestrians and cars to show its accuracy and efficiency.

The rest of this paper is organized as follows: section 2 formulates the tree structured classifier model; section 3 gives our learning algorithm; section 4 shows some experimental results; and conclusions and discussions are in the last section.

2. Tree Structured Classifier

First we formally define a tree structured classifier. The input of such a classifier is an image sample, $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}$ is the sample space, and the output is a vector. Let

$$H(\mathbf{x}) = [H_1(\mathbf{x}), \ldots, H_C(\mathbf{x})] \tag{1}$$

be the tree classifier, where C is the number of subcategories, called *channels*. Each channel is an ensemble classifier:

$$H_k(\mathbf{x}) = \sum_{t=1}^{T} h_{k,t}(\mathbf{x}), k = 1, \ldots, C \tag{2}$$

where $h_{k,t}$ is called *weak* classifier, and T is the number of weak classifiers, *i.e*. the depth of the tree. The input of a weak classifier is an image sample $\mathbf{x}$ and the output is a real-valued number, whose sign is the class prediction (positive for object and negative for non-object). Each weak classifier is built based on one image feature f. Weak classifiers at the same level may share the same feature. Fig.1 gives an illustration of the tree structure. The tree shown has three levels and three channels. At the first level, all the weak classifiers of the three channels share the same feature, indicated by the red box, though they may have different classification functions. At the second level, the third channel stops sharing feature with the other two channels; this is called a splitting operation. At the third level, the first and second channels split. Feature sharing has proven to be an efficient strategy for multi-class object detection [13] and multi-view object detection [8, 11]. The underlying assumption is that although different views of the objects look quite different, they may share some common visual features.

To integrate with cascade decision strategy, a threshold $b_{k,t}$ is learned for each weak classifier $h_{k,t}$. Let $H_{k,t}$ be the partial sum of the first t weak classifiers of channel k, *i.e*.

$$H_{k,t}(\mathbf{x}) = \sum_{j=1}^{t} h_{k,j}(\mathbf{x}) \tag{3}$$

A sample is accepted by channel i, if and only if

$$\forall t \in \{1, \ldots, T\}, H_{k,t} > b_{k,t} \tag{4}$$

A sample is classified as an object, if and only if it is accepted by at least one channel.

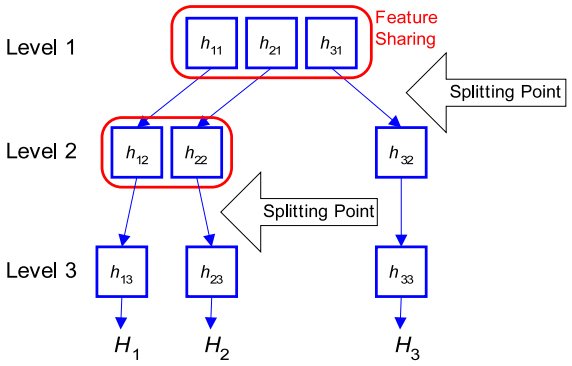


Figure 1. Example of tree structure.

3. Learning Algorithm

Our learning algorithm is based on the real AdaBoost algorithm proposed by Schapire and Singer [19]. An image feature can be seen as a function from the image space to a real valued range, $f : \mathcal{X} \mapsto [0, 1]$. The weak classifier based on f is a function from the image space $\mathcal{X}$ to a real valued object/non-object classification confidence space. Given a labeled sample set $S = \{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x} \in \mathcal{X}$ is the image patch, and $y_i = \pm 1$ is the class label of $\mathbf{x}$, in real AdaBoost we first divide the sample space into several parts:

$$\mathcal{X} = \bigcup_{j=1}^{n} X_j \tag{5}$$

where n is the size of the partition. Then the weak classifier h is defined as a piecewise function:

$$\text{if } \mathbf{x} \in X_j, h(\mathbf{x}) = \frac{1}{2} \ln \left(\frac{W_+^j + \varepsilon}{W_-^j + \varepsilon} \right), j = 1, \ldots, n \tag{6}$$

where ε is a smoothing factor [19], and $W_\pm$ is the probability distribution of the feature value for positive/negative samples, implemented as a histogram:

$$W_\pm^j = P\left(\mathbf{x} \in X_j, y = \pm 1\right), j = 1, \ldots, n \tag{7}$$

In practice, following [11] the partition of the sample space is achieved by dividing the feature space into n equal sized sub-ranges, *i.e.*

$$X_j = \left\{ \mathbf{x} \middle| f(\mathbf{x}) \in \left[\frac{j-1}{n}, \frac{j}{n} \right) \right\}, j = 1, \ldots, n \tag{8}$$

3.1. Splitting sample space

As our purpose is to construct tree classifiers without a predefined intra-class sub-categorization, we need to answer two questions: when is the sub-categorization necessary and how to divide the sample space? In real AdaBoost [19], the classification power of the weak classifier is measured by a Z value:

$$Z = 2 \sum\nolimits_j \sqrt{W_+^j W_-^j} \tag{9}$$

When $W_\pm$ is normalized, $Z \in [0, 1]$ is the Bhattacharyya distance between the distributions of the object and non-object classes. The smaller Z is, the more discriminative the weak classifier is. At each boosting round the weak classifier with the smallest Z is selected. Due to the cascade decision strategy, the classifier focuses on the hard part of the sample space gradually. Thus it becomes more and more difficult to find weak classifiers with small Z value during the boosting procedure. Fig.2 shows the evolution of the Z values of the selected weak classifiers. When Z gets very close to 1, its contribution to the whole classifier is little, or in other words the current problem is beyond its ability. This suggests a division. In practice we set a threshold θ_Z, if the Z values of three consecutive weak classifiers are all larger than θ_Z, the splitting operation is triggered.

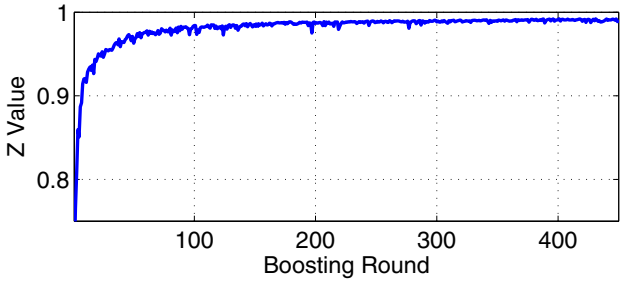


Figure 2. Evolution of Z value.

Because our method is a discriminative approach based on image features, we use a bottom-up splitting strategy to facilitate the classification, instead of top-down. We argue that the features selected by boosting form a good descriptor of the object class, and they are selected for the classification task directly. Hence clustering based on these features is a better way to divide the sample space than those based on domain knowledge or posterior probability. Later in section 4.1 we will show the advantage of our splitting method. Suppose up to the splitting point, K features have been selected, then the vector $\mathbf{f} = \{f_1, \ldots, f_K\}$ is fed into a unsupervised clustering algorithm to divide the sample space into two parts. In practice, we use the standard k-mean algorithm with Euclidean distance. Although there are many other clustering algorithms, we found this serves our purpose well.

3.2. Retraining with sub-categorization

During the training procedure, without predefined sub-categorization, one leaf node of the tree always corresponds to one single channel. Suppose the current tree has t levels and c channels. When a leaf node $h_{k,t}$ splits, two branches with a new channel are generated, $h_{k,t+1}$ and $h_{c+1,t+1}$.

- Given the initial sample set $S = S_{+1} \cup S_{-1} = \{(\mathbf{x}_i, +1\} \cup \{(\mathbf{x}_i, -1)\}$, and a negative images set;
- Set the algorithm parameters: the maximum weak classifier number T, the positive passing rates $\{P_t\}_{t=1}^{T}$, the target false alarm rate F, the maximum channel number C, the splitting threshold θ_Z, and the threshold for bootstrapping θ_B;
- Initialize the weights $D_0(\mathbf{x}) = \frac{1}{\|S\|}$ for all samples, the current false alarm rate of the current channel $F_{1,0} = 1$, the current channel number $c = 1$, and $t = 0$;
- Construct the weak classifier pool, $\mathcal{H}$, from the image features;
- while $t < T$ do
  1. For all channels, $k = 1, \ldots, c$, do
     (a) If $F_{k,t} < F$, continue;
     (b) For all weak classifier in $\mathcal{H}$, learn the classification function from the sample sets $S_{\pm k}$ by Equ.6;
     (c) Compute the Z values for all weak classifiers by Equ.9, select the weak classifier with the smallest Z, denoted by $h_{k,t}$ and its corresponding feature $f_{k,t}$;
     (d) Update sample weights by
     $$D_{t+1}(\mathbf{x}) = D_t(\mathbf{x}) \exp\left[-y h_{k,t}(\mathbf{x})\right], \forall \mathbf{x} \in S_{\pm k} \tag{10}$$
     and normalize D_{t+1} to a probability distribution;
     (e) Select the threshold $b_{k,t}$ for the partial sum $H_{k,t}$, so that a portion of P_t positive samples are accepted; and reject as many negative samples as possible;
     (f) Remove the rejected samples from $S_{\pm k}$. If the remaining negative samples are less than θ_B percent of the original, recollect S_{-k} by bootstrapping on the negative image set and update $F_{k,t}$;
     (g) If Z of $h_{k,t}$, $h_{k,t-1}$, and $h_{k,t-2}$ are all larger than θ_Z, and $c < C$, do splitting and retraining
        i. $\forall \mathbf{x} \in S_{+k}$, compute the feature descriptor $\mathbf{f}(\mathbf{x}) = [f_{k,1}(\mathbf{x}), \ldots, f_{k,t}(\mathbf{x})]$;
        ii. Based on $\mathbf{f}(\mathbf{x})$, do k-mean clustering on S_{+k} to divide it into two parts, and assign the labels k and $c+1$ to the two new sub-categories, *i.e.* $S_{+k} \rightarrow S_{+k} \cup S_{+(c+1)}$;
        iii. Retrain all the previous weak classifiers for channel k with the algorithm in Fig.4;
        iv. $c++$;
  2. $t++$;
- Output $\{h_{k,t}, b_{k,t}\}$ as the tree structured classifier for detection.

Figure 3. Algorithm of learning tree structured classifier. In our experiments, $T = 500$, $F = 10^{-6}$, $C = 8$, $\theta_Z = 0.985$ and $\theta_B = 75\%$. The setting of $\{P_t\}$ is similar to the original cascade's layer acceptance rates. The cascade is divided into 20 segments, the lengthes of which grow gradually. The weak classifiers at the end of the segments have positive passing rate of 99.8%, and the other weak classifiers have passing rate of 100.0%.

- Given the sample sets of the new channels S_{+k} and $S_{+(c+1)}$;
- Inherit all the parameters from the main algorithm in Fig.3;
- Randomly collect the initial negative sample sets for the new channels, S_{-k} and $S_{-(c+1)}$;
- Reset the weights $\forall \mathbf{x} \in S_{+k} \cup S_{+(c+1)}$, $D_0(\mathbf{x}) = \frac{1}{\|S\|}$;
- For $t' = 1$ to t do
  1. Split the original $h_{k,t'}$ into two, $h_{k,t'}$ and $h_{c+1,t'}$, which share the same feature, *i.e.* $f_{k,t'} = f_{c+1,t'}$
  2. Retrain $h_{k,t'}$ with the following steps:
     (a) Learn the classification function from the new sample sets $S_{\pm k}$ by Equ.6;
     (b) Update the sample weights by Equ.10;
     (c) Select the threshold $b_{k,t'}$ for the partial sum $H_{k,t'}$, so that a portion of $P_{t'}$ positive samples are accepted; and reject as many negative samples as possible;
     (d) Remove the rejected samples from $S_{\pm k}$. If the remaining negative samples are less than θ_B percent of the original, recollect S_{-k} by bootstrapping on the negative image set;
  3. Retrain $h_{c+1,t'}$ with $S_{\pm(c+1)}$ by the same steps for $h_{k,t'}$.

Figure 4. Retraining of classification functions with intra-class sub-categorization.

The new sub-categorization information is sent upstream to all ancestors of $h_{k,t+1}$ and $h_{c+1,t+1}$ to refine their classification functions. For example, $h_{k,t}$ changes to a pair $\{h_{k,t}, h_{c+1,t}\}$. The whole boosting procedure is then rerun from the beginning for the two changed channels, k and $c+1$. Different from standard boosting, in the retraining we do not do feature selection and only retrain the classification functions, *i.e.* $h_{k,t}$ and $h_{c+1,t}$ share the original feature for $h_{k,t}$ and their classification functions are retrained on the samples of the new sub-categories. Later in section 4.2, we show that the retrained weak classifiers have better performance.

Our complete learning algorithm is given in Fig.3 and Fig.4. Note that learning of the cascade decision strategy is integrated with the boosting procedure. This enables us to represent the learning algorithm as one boosting procedure, instead of several stages like in [18]. The main new parameter compared to the previous cascade learning methods is the splitting threshold θ_Z, which is set based on experience. We find that the resulting classification accuracy is not very sensitive to this parameter. The output classifier of our learning algorithm has the same structure as that in [11], while our algorithm does not require the predefined intra-class sub-categorization.

4. Experimental Results

We apply the proposed method to the problem of pedestrian detection and car detection. Besides an end-to-end comparison, evaluations of the individual modules of the system are helpful to understand the different parts of the method. We do experiments to compare our design choices with some other existing methods. Following our previous work [12], we use *edgelets* which are for silhouette patterns as the image features.

In section 4.1, 4.2, and 4.3, we evaluate our method on the pedestrian detection problem and compare with previous methods. For pedestrian detection, there have been several data sets publicly available, *e.g.* the INRIA pedestrian set [9][1]. The INRIA set covers all view points and a large variation of poses. It contains 2,478 positive samples and 1,218 negative images for training, and 1,128 positive samples and 453 negative images for testing. As the INRIA set contains segmented samples, it is good for comparison of the learning algorithms. However, for multi-view pedestrians, there is no public set for evaluating the *detection* performance of the whole system. Hence we collect our own multi-view pedestrian test set. Following the our previous work [12], the pedestrian samples in our experiments are resized to 24×58 pixels.

In section 4.4, we apply our method to multi-view car detection. For cars, there are several public sets, *e.g.* the UIUC car set[2] [15], which include a training set, a single-scale test set, and a multi-scale test set. But all the existing car image sets are for a single view point, *e.g.* the UIUC set contains only profile view cars. Hence, we collect a training set for multi-view cars from the MIT street scene images[3] [16]. This training set contains 4,000 car samples of various models and different view points. The car samples are resized to 64×32 pixels.

[1] http://pascal.inrialpes.fr/data/human/

[2] http://l2r.cs.uiuc.edu/~cogcomp/Data/Car/

[3] http://cbcl.mit.edu/software-datasets/streetscenes/

4.1. Comparison of splitting strategies

First, we compare three splitting strategies, Tu's posterior probability based method [10], a pure random splitting method, and our image feature based k-mean. In [10], a posterior probability is computed by

$$q(\pm 1|\mathbf{x}) = \frac{\exp\{\pm 2H(\mathbf{x})\}}{1 + \exp\{\pm 2H(\mathbf{x})\}} \tag{11}$$

When the classification is confident, *i.e.* $q(\pm 1|\mathbf{x}) - \frac{1}{2} > \epsilon$, where ϵ defines a gray "not sure" area, the sample is sent to left or right branch according to the sign. Samples in the gray area are sent to both left and right branches. As the original work of Tu [10] is designed for general classification problem, they set the threshold to 0.5. However, for a detector with a cascade decision strategy, the probability of most of the positive samples is close to 1, so we use a threshold of 0.85 in our experiment. The pure random splitting strategy is to divide the sample set into two equal sized sub-sets randomly.

In order to evaluate the splitting strategy, we deactivate the retraining module and force the tree to split at level 50 and level 100, and the maximum number of weak classifier is set to 150. This experiment is done using the training samples of the INRIA set. Note that we do not implement the whole algorithm of Tu [10], instead we only use their splitting strategy in our framework. Intuitively, the better is the splitting, the more powerful should the weak classifier selected later on be. The convergence speed of the learning procedure depends on the hardest sub-category, hence we trace the maximum Z value of the weak classifiers at each tree level and use the evolution curve of the maximum Z value to measure the performance of the splitting strategies. Fig.5 shows the curves of maximum Z value, and Fig.6 shows the resulting tree structures by the posterior probability based strategy and our image feature based k-mean strategy.

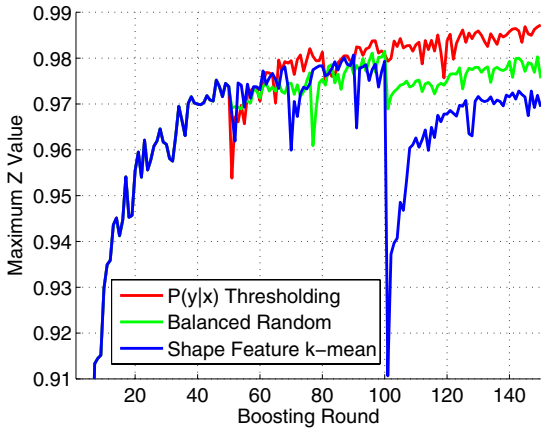


Figure 5. Comparison of splitting strategies.

From Fig.5, we can see that the feature based splitting method achieves the best effect on facilitating classification. Another interesting result is that even random splitting method is better than the posterior probability based method. This is because the probability based method generates a highly unbalanced tree, while both the other two methods result in balanced trees, see Fig.6. Because of the asymmetric property of the detection problem and the cascade decision strategy, the value of $p(+1|\mathbf{x})$ keeps increasing for all samples. The sample distribution along the posterior probability tends to be a single peak gaussian with small variance, and the peek is close to one. No matter what splitting threshold used, it is unlikely to get balanced divisions on both positive and negative sample spaces. This suggests that the scalar valued posterior probability may not be a good clustering criteria for object detection tasks.

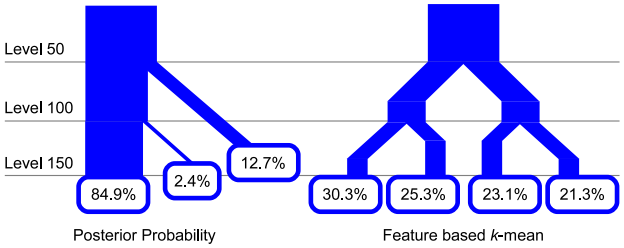


Figure 6. Tree structures from different splitting strategies. (The number in the box gives the percentage of samples belonging to that branch. The tree from random splitting strategy is similar to that from feature based k-mean.)

4.2. Comparison on data without predefined sub-categorization

Next, we compare our method with some previous work [1, 2, 9] on the INIRA set which does not have view/pose labels. We learn the classifiers with the training samples of the INRIA set, and evaluate on its test samples. For our method, three runs are done with different settings. In the first run, we set the maximum number of channels to one ($C = 1$), which reduces the classifier to a cascade structure. In the second run, we set $C = 4$ and disable the retraining module. In the last run, we set $C = 4$ and enable the retraining module. The other parameters of these three runs are same. Fig.8 gives the ROC curves.

From the results, it can be seen that for our method, four channel no retraining CBT classifier outperforms the cascade one; while the four channel with retraining CBT classifier outperforms both of them. This shows the effectiveness of the splitting and retraining modules. Comparing to the existing work, ours is comparable to [1, 9], while the very recent shapelet based method by Sabzmeydani and Mori [2] dominates all the others. (Another recent work not included in Fig.8 is the HOG based boosting method by Zhu *et al*. [6], performance of which is similar to that of [9].) However, [1, 2, 9] all focus on developing stronger features, while our work focuses on the classifier structure and training. These are two complementary directions. Fig.7 shows some failure examples by our method.

Figure 7. Missed human examples by our method with a false alarm rate of 10^{-4}.

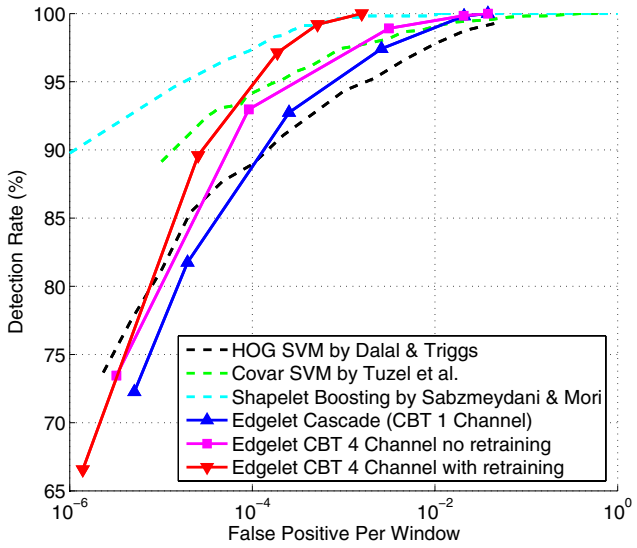


Figure 8. Evaluation on the INRIA set [9].

4.3. Comparison on data with view-based sub-categorization

In [11], the intra-class sub-categorization of face samples is based on view point, which is a common solution for multi-view object detection. In this experiment, we compare our unsupervised sub-categorization based method with the predefined sub-categorization based method. We divide the pedestrian class into three view-based sub-categories, left profile, frontal/rear, and right profile. As the INRIA set does not have view point information, we use our own collection from the Internet for this experiment. Our training set contains 1,000 positive samples for left profile, 1,000 positive for frontal/rear, 1,000 positive for right profile, and 7,000 negative images without humans. For the view based classifier, we use Vector Boosting [11] to train a three channel cascade as root. When the false alarm rate of the root reaches 10^{-4}, we stop the feature sharing and split it into three branches, each of them then reaches a false alarm rate of 10^{-6}. For our method, we set $C = 8$, however the resulting tree ends up with only four channels. To evaluate the detection performance, we collect a test set containing 100 images with 232 multi-view pedestrians.[4] Fig.9

[4] http://iris.usc.edu/~bowu/DatasetWebpage/dataset.html

shows the precision-recall curves of the two tree classifiers on this test set.

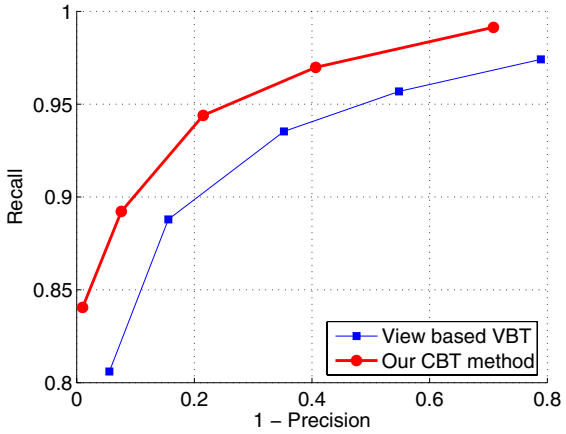


Figure 9. Evaluation on multi-view pedestrian test set (100 images with 232 pedestrians).

From the results it can be seen that the CBT detector dominates the view-based VBT detector. One possible reason is that the view-based sub-categorization is not optimal for the pedestrian class as the articulation is also an important factor that affects the appearance. Different from the domain-knowledge based sub-categorization, our clustering method is based on the discriminative features selected directly for the classification task. We find no obvious physical meanings of the learned sub-categories.

Another advantage of our CBT method is that it always generates balanced trees, which are more efficient than unbalanced ones. In the view based VBT detector we learned, the branches for profile views are much longer than the branch for frontal/rear view, which means the classification difficulty is not equally distributed in the view based sub-categories. Finally Fig.10(a) shows some examples of successful detection for multi-view pedestrians.

4.4. Multi-view car detection

To show the generality of the proposed method, we apply it to the problem of multi-view car detection. The 4,000 car samples collected from the MIT street scene image set are taken as our positive training set, the 7,000 negative images in section 4.3 are also used here as the negative set. A CBT detector with four channels are learned for multi-view cars. To compare with previous methods, we learn a cascade detector from the training data of UIUC car set [15], which only contains profile view cars, and test the CBT detector for multi-view cars and the cascade detector for side view cars on the UIUC single scale set, which has 170 images with 200 cars, and the multi-scale test set, which has 108 images and 139 cars. Comparison results are given in Table.1. It can be seen that our multi-view CBT detector has performance comparable to the state-of-the-art single-view approach on a particular view point. We show some detection results for multi-view car detection in Fig.10(b). Compared to the previous methods on multi-view car detection, *e.g.* [4], our method covers a similar range of view points but with more variations of car models.

Method	Single-scale	Multi-scale
Leibe *et al.* [14]	97.5%	-
Todorovic & Ahuja [7]	$\sim$ 86.5%	-
Mutch & Lowe [5]	99.94%	90.6%
Single view cascade	97.5%	93.5%
Multi-view CBT	97.5%	92.8%

Table 1. Detection equal-precision-recall rates on the UIUC car image set.

Our program is coded in C++ using OpenCV functions without any code level optimization or parallel computing. The experiments in section 4 are done with a 2.8G Hz 32-bit Pentium PC. Our CBT method takes about one day to learn one classifier; this is slower than the learning of SVM classifiers but much faster than that of VBT classifiers. The detection speed depends on the size of the object and the image. Usually it takes only few hundreds milliseconds for one image.

5. Conclusion and Discussion

We described a method to learn tree structured object detectors automatically, without predefined intra-class sub-categorization. We divide the sample space by unsupervised clustering based on image features selected by boosting and refine the classification functions of the ancestors by the sub-categorization information of their children in the tree. The experimental results show that our method outperforms the previous methods.

In this work, we learn our detector based on edgelet features [12], which is one type of shape features. There are many other candidates, such as SIFT [20], HOG [9], Covariance Descriptor [1], shapelet [2], and Haar [18]. Our learning algorithm does not limit the type of features used. New features could be integrated to the framework easily.

Acknowledgements: The authors would like to thank Dr. Mori and Dr. Tuzel for kindly providing their result data for comparison. This research was funded, in part, by the U.S. Government VACE program.

References

[1] O. Tuzel, F. Porikli, and P. Meer. Human Detection via Classification on Riemannian Manifolds. CVPR 2007. 6, 7

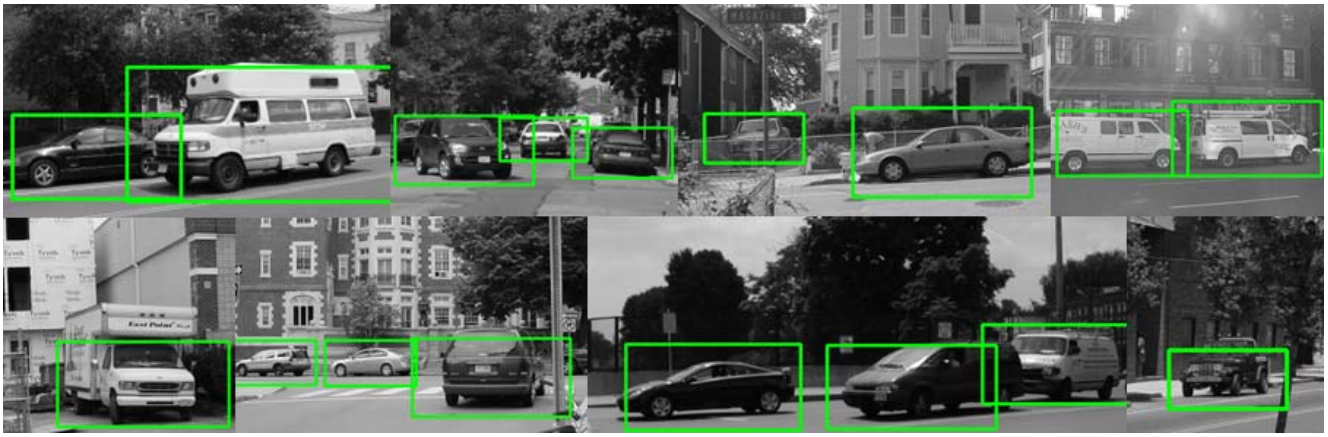
[2] P. Sabzmeydani, and G. Mori. Detecting Pedestrians by Learning Shapelet Features. CVPR 2007. 6, 7

[3] E. Seemann, B. Leibe, and B. Schiele. Multi-Aspect Detection of Articulated Objects. CVPR 2006. 2

[4] Y. Shan, F. Han, H. Sawhney, and R. Kumar. Learning Examplar-Based Categorization for the Detection of Multi-View Multi-Pose Objects. CVPR 2006. 2, 7

[5] J. Mutch, and D. Lowe. Multiclass Object Recognition with Sparse, Localized Features. CVPR 2006. 7

(a) Example results of pedestrian detection



(b) Example results of car detection

Figure 10. Example detection results. (These examples are not included in our training data.)

[6] Q. Zhu, S. Avidan, M.-C. Yeh, and K.-T. Cheng. Fast human detection using a cascade of Histograms of Oriented Gradients. CVPR 2006. 6

[7] S. Todorovic, and N. Ahuja. Extracting Subimages of an Unknown Category from a Set of Images. CVPR 2006. 7

[8] Y.-Y. Lin, and T.-L. Liu. Robust Face Detection with Multi-Class Boosting. CVPR 2005. 2

[9] N. Dalal, and B. Triggs. Histograms of Oriented Gradients for Human Detection. CVPR 2005. 2, 5, 6, 7

[10] Z. Tu. Probabilistic Boosting-Tree: Learning Discriminative Models for Classification, Recognition, and Clustering. ICCV 2005. 1, 2, 5

[11] C. Huang, H. Ai, Y. Li, and S. Lao. Vector Boosting for Rotation Invariant Multi-View Face Detection. ICCV 2005. 1, 2, 3, 5, 6

[12] B. Wu, and R. Nevatia. Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors. ICCV 2005. 1, 5, 7

[13] A. Torralba, K.P. Murphy, and W.T. Freeman. Sharing Features: Efficient Boosting Procedures for Multiclass Object Detection. CVPR 2004. 2

[14] B. Leibe, A. Leonardis, and B. Schiele. Combined Object Categorization and Segmentation with an Implicit Shape Model. Workshop on Statistical Learning in Computer Vision, in conjunction with ECCV 2004. 7

[15] S. Agarwal, A. Awan and D. Roth. Learning to detect objects in images via a sparse, part-based representation. PAMI, 26(11):1475–1490, November 2004. 5, 7

[16] B. Leung. Component-based Car Detection in Street Scene Images. Master's Thesis, EECS, MIT, 2004. 5

[17] M. Jones and P. Viola. Fast Multi-View Face Detection. TR2003-96 1

[18] P. Viola and M. Jones. Rapid Object Detection Using a Boosted Cascade of Simple Features. CVPR 2001. 1, 5, 7

[19] R. E. Schapire and Y. Singer. Improved Boosting Algorithms Using Confidence-rated Predictions. Machine Learning 1999. 3

[20] D. G. Lowe. Object recognition from local scale-invariant features. ICCV 1999. 7

[21] Y. Freund, and R. E. Schapire. Experiments with a New Boosting Algorithm. Machine Learning 1996. 1