Efficient Object Detection Using Cascades of Nearest Convex Model Classifiers

Hakan Cevikalp
Eskisehir Osmangazi University
Meselik Kampusu, 26480, Eskisehir Turkey
hakan.cevikalp@gmail.com

Bill Triggs
Laboratoire Jean Kuntzmann
B.P. 53, 38041 Grenoble Cedex 9, France
Bill.Triggs@imag.fr

Abstract

An object detector must detect and localize each instance of the object class of interest in the image. Many recent detectors adopt a sliding window approach, reducing the problem to one of deciding whether the detection window currently contains a valid object instance or background. Machine learning based discriminants such as SVM and boosting are typically used for this, often in the form of classifier cascades to allow more rapid rejection of easy negatives. We argue that "one class" methods – ones that focus mainly on modelling the range of the positive class – are a useful alternative to binary discriminants in such applications, particularly in the early stages of the cascade where one-class approaches may allow simpler classifiers and faster rejection. We implement this in the form of a short cascade of efficient nearest-convex-model one-class classifiers, starting with linear distance-to-affine-hyperplane and interior-of-hypersphere classifiers and finishing with kernelized hypersphere classifiers. We show that our methods have very competitive performance on the Faces in the Wild and ESOGU face detection datasets and state-of-the-art performance on the INRIA Person dataset. As predicted, the one-class formulations provide significant reductions in classifier complexity relative to the corresponding two-class ones.

1. Introduction

Object class detection is an important computer vision task in which all instances of a given generic object class that occur in an image must be recovered and labeled with their correct image positions and scales. It is difficult owing to the highly variable shape and appearance of common object categories, changing scales, view-points and lighting conditions, complex backgrounds, occlusion and clutter. The methods that currently dominate the field are based on scanning the image at multiple scales with window-level object / non-object classifiers that use machine learning discriminants such as Support Vector Machines over high-dimensional visual feature sets [7].

Both the feature set and the classifier are critical for obtaining good performance. Here we concentrate on the classifier. We introduce a novel short cascade approach that uses "one-class" component classifiers based on simple convex geometric models and graduated nonlinearity. Geometric one-class approaches prioritize the accurate and efficient approximation of the feature space regions occupied by the positive object class over the explicit discrimination of positives from negatives. Particularly in the earlier stages of the cascade, this simplifies the component classifiers and allows early rejection of easy negatives. Specifically, our method combines distance-to-affine-hyperplane, linear hypersphere and kernelized hypersphere classifiers.

We also enhance some existing one-class classification software to handle large-scale problems and introduce a new face detection dataset. The well-established MIT+CMU face dataset [21] is somewhat dated in the sense that it includes only a limited number of images, and that these are grayscale with relatively low resolutions. The majority of the faces in the newer Faces in the Wild dataset[1] appear in the middle of the image with similar scales, limiting its value as a test set for multiscale face detectors (it is principally a face recognition dataset, not a face detection one). We therefore developed ESOGU Faces, a new frontal face detection dataset containing 285 higher-resolution color images of complex real-world scenes taken under a wide range of different illumination conditions.

2. Previous Work

Regarding feature sets, early detectors used raw pixel values [22], wavelets [19], edges [3], and Gabor filter responses [23]. More recently, histogram based features have become very popular owing to their performance and efficiency. Many of these are based on oriented image gradients, including SIFT [16], SURF [4], Histogram of Oriented Gradients (HOG) [6], PHOG [26], Generalized Shape Context [5] and Local Edge Orientation Histograms [15]. Oth-

[1] http://vis-www.cs.umass.edu/lfw

ers are based on local patterns of qualitative graylevel differences, including Local Binary Patterns (LBP) [1,28] and Local Ternary Patterns (LTP) [24]. The best feature set depends on the application and new ones are being developed all the time. Current detectors often combine several feature sets for better results, either simply concatenating them to form an extended feature vector [10], or finding optimal combination coefficients at the learning stage [26].

Regarding the decision rule, most methods reduce the detection problem to binary classification, *i.e*. determining whether the detector window currently contains a correctly framed true class instance or something else (background, a partial or incorrectly framed instance, another class, *etc*.). Machine learning classifiers ranging from nearest neighbors to neural networks, convolution neural networks, probabilistic methods and classification trees have been used, but two approaches have received much of the attention owing to their interesting properties: boosting based cascades, and Support Vector Machines. Viola & Jones [27] produced a very efficient face detector by using AdaBoost to train a cascade of pattern-rejection classifiers over rectangular wavelet features. Each stage of the cascade is designed to reject a considerable fraction of the negative cases that survive to that stage, so most of the windows that do not contain faces are rejected early in the cascade with comparatively little computation. As the cascade progresses, rejection typically gets harder so the single-stage classifiers grow in complexity.

Although cascades give excellent results for real-time face detection, Support Vector Machine (SVM) classifiers are currently a more common choice for more general object detection under less stringent time constraints [10,9,6,26,2]. Linear SVM's are usually preferred for their simplicity and speed, although it is well-established that kernel SVM's typically give higher accuracy at the cost of greatly increased computational complexity [26]. For this reason, several state-of-the art methods use short cascades in which the early stages use linear SVM's to reject most of the negative windows quickly while the later stages use nonlinear SVM's to make the final decisions [10,26].

Several previous detectors have used one-class formulations such as the Support Vector Data Description (SVDD) method of Tax and Duin [25], which approximates classes with hyperballs (the interiors of hyperspheres) in feature space. Jin *et al*. [13] use a kernelized hypersphere model for face detection. However to reduce the computational complexity, they first divide the putative face window into 9 blocks using heuristic rules such as the eye regions being darker than the cheeks and the bridge of nose, *etc*., only applying the nonlinear classifier if the region passes all of these tests. Their method thus applies only to face detection. Mele & Maver [18] use linear hypersphere classifiers to detect specific shapes in binary segmented images, but their approach is not applicable to general object detection in color or grayscale images. In contrast, our method can be used to detect any more or less rigid object class in natural images, and it is significantly faster than nonlinear two class SVM based detectors while maintaining comparable overall accuracy.

3. Our Approach

In object localization any sample that does not belong to the object class is considered to be background, so in feature space, the class samples typically lie in specific regions surrounded by a diffuse sea of background samples. Given that such backgrounds are defined negatively (as anything at all that is not a well-framed class instance), discriminative training methods typically need to process very large numbers of negative training samples to represent them well. This explains both their need for multiple cycles of search for hard negatives and retraining [6], and the extremely unbalanced non-class to class ratios that result from this. We argue that this is counterproductive, and that (at least in the early stages of the cascade) it is preferable to concentrate on modeling the extent of the positive class well, discarding any sample that does not conform to this model as an 'easy' negative and postponing the more nuanced decisions until later.

To achieve this we treat each stage of the object detection cascade as a one-class nearest-convex-model classification problem, not a two-class discrimination problem. For each stage of the cascade we would ideally use the positive training samples alone to learn a convex approximation to the region occupied by the class in feature space, then classify samples as possible-class or background by thresholding their feature-space distances to the convex model. In practice, we find that it is useful to include some background information from the negative samples in each stage, while still retaining "one-class" style models and philosophy. Below we will use either affine hyperplanes or bounding spheres for our convex cascade-stage models because it is very efficient to find distances to these, thus potentially allowing real-time performance. However many other models are possible at the cost of more expensive distance computations, including lower-dimensional affine subspaces, convex hulls, hyper-disks and hyper-ellipsoids, etc.

In this paper we will focus on a particular form of cascade in which each stage uses a different kind of nearest convex model classifier, as illustrated in Fig. 1. Our cascades have three stages graded by the computational complexity of the model. The first stage is a linear classifier that uses an intersection of affine hyperplanes to approximate the class. This is computationally efficient and also easy and fast to train, even for large training sets. Its goal is to reject as many of the background samples as possible, while still passing almost all of the class samples to the next stage. The

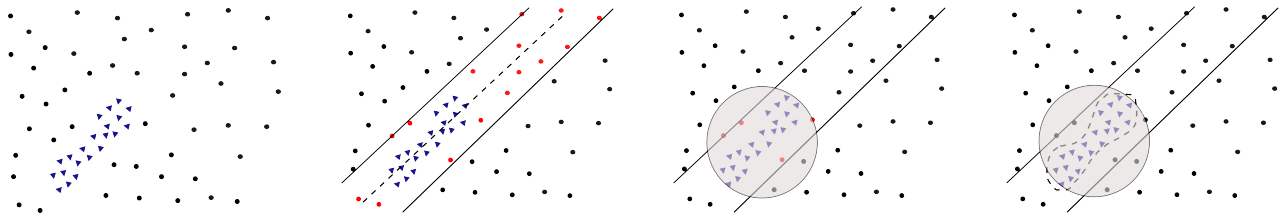

Figure 1. (Best viewed in color). An illustration of pruning in our proposed three-stage cascade. (a) Input data: points from the object (positive) and background (negative) classes are shown respectively as blue triangles and black dots. (b) In the first stage of the cascade, the positive samples are bounded with a series of hyperplane shaped slabs (the dashed line and its two borders). Samples outside the slabs are classified as negatives and rejected. The background examples that survived this stage are shown as red dots. (c) The second stage of the cascade is a linear-space one-class classifier that approximates the object class region with a bounding hypersphere. Most of the false positives that survive the first stage are discarded here. (d) The final stage of the cascade is a kernelized one-class classifier that approximates the positive region more accurately and makes the final decision.

second stage is a linear SVDD classifier [25], *i.e.* one based on a hypersphere model in the (non-kernelized) input space. This is equally fast to run, but somewhat more expensive to train as it uses a maximum-margin formulation rather than simple linear fitting. It turns out to be complementary to the first stage, rejecting most of the false positives that stage 1 passes. The third stage of the cascade makes the final decisions, using a kernelized hypersphere model to approximate the object class. This is slower, but it only needs to test the small number of positives and difficult negatives passed by the first two stages. We now present each of these three stages in detail.

3.1. Linear Hyperplane Approximation

The first stage of our cascade tests the distance of the sample to a series of affine hyperplanes. Let $\mathbf{x}$ be the sample's feature vector and let $\mathbf{w}_1^\top \mathbf{x} + b_1 = 0$ be the equation of the first hyperplane, where $\mathbf{w}_1$ is a unit vector of feature weights and b_1 is a bias. We reject samples for which $|\mathbf{w}_1^\top \mathbf{x} + b_1| > \tau_1$, where τ_1 is a threshold determined by cross-validation. For the surviving samples, we find the orthogonal complement $\mathbf{x}_1 = \mathbf{x} - \mathbf{w}_1 (\mathbf{w}_1^\top \mathbf{x})$, and pass it on to the next hyperplane in the series for testing. This continues throughout the series, at each stage testing the distance of the current vector to the current hyperplane and passing on its orthogonal complement if it survives.

Let $\mathbf{X}_+$ and $\mathbf{X}_-$ be matrices whose rows are the training samples of respectively the object and background classes. Let $\mathbf{e}_+$ and $\mathbf{e}_-$ be corresponding column vectors of ones, and for convenience define extended matrices $\bar{\mathbf{X}}_+ = [\mathbf{X}_+ \;\; \mathbf{e}_+]$ and $\bar{\mathbf{X}}_- = [\mathbf{X}_- \;\; \mathbf{e}_-]$. To train the method, the simplest one-class approach would be to find the best least-squares fit to the positive data

$$\underset{\mathbf{w}_1, b_1, \|\mathbf{w}_1\|=1}{\arg\min} \|\mathbf{X}_+ \mathbf{w}_1 + \mathbf{e}_+ b_1\|^2 = \underset{\mathbf{z}}{\arg\min} \frac{\mathbf{z}^\top \mathbf{G}\, \mathbf{z}}{\|\mathbf{w}_1\|^2} \quad (1)$$

where $\mathbf{z} = \begin{pmatrix} \mathbf{w}_1 \\ b_1 \end{pmatrix}$ and $\mathbf{G} = \bar{\mathbf{X}}_+^\top \bar{\mathbf{X}}_+$, passing the orthogonal complement $\mathbf{X}_+ (\mathbf{I} - \mathbf{w}_1 \mathbf{w}_1^\top)$ to the next stage of the series as training data. Instead we use a background-sensitive fit [17], minimizing the regularized Rayleigh quotient

$$\underset{\mathbf{w}_1, b_1, \|\mathbf{w}_1\|=1}{\arg\min} \frac{\|\mathbf{X}_+ \mathbf{w}_1 + \mathbf{e}_+ b_1\|^2}{\|\mathbf{X}_- \mathbf{w}_1 + \mathbf{e}_- b_1\|^2 + \delta\,(\|\mathbf{w}_1\|^2 + b_1^2)}, \quad (2)$$

which can be re-expressed as

$$\underset{\mathbf{z}}{\arg\min} \frac{\mathbf{z}^\top \mathbf{G}\, \mathbf{z}}{\mathbf{z}^\top \mathbf{H}\, \mathbf{z}} \quad (3)$$

where $\mathbf{H} = \bar{\mathbf{X}}_-^\top \bar{\mathbf{X}}_- + \delta \mathbf{I}$ and δ is a user-set regularization constant. Again the orthogonal complements of $\mathbf{X}_+$ and $\mathbf{X}_-$ are passed to the next hyperplane as training data. The solution of this problem reduces to finding the smallest-λ eigenvector of the generalized eigenproblem $\mathbf{G}\, \mathbf{z} = \lambda\, \mathbf{H}\, \mathbf{z}$ and renormalizing it to find $\mathbf{w}_1, b_1$.

3.2. Linear Hypersphere Approximation

The second stage of the cascade consists of a single linear SVDD classifier [25]. SVDD uses bounding hyperspheres to approximate classes. As Fig. 1 suggests, the hypersphere classifiers turn out to complement the preceding hyperplane ones well, rejecting most of the false positives that survive the first stage.

The bounding hypersphere of a point set $\{\mathbf{x}_i | i = 1 \ldots n\}$ is characterized by its center $\mathbf{c}$ and radius r. These can be found by solving the quadratic programming problem

$$\underset{\mathbf{c},\, r \geq 0,\, \boldsymbol{\xi} \geq 0}{\arg\min} \left(r^2 + \gamma \sum_i \xi_i \right) \quad \text{s.t.} \;\; \|\mathbf{x}_i - \mathbf{c}\|^2 \leq r^2 + \xi_i, \quad i = 1, \ldots, n, \quad (4)$$

or its dual

$$\begin{aligned} \arg\min_{\boldsymbol{\alpha}} \quad & \left(\sum_{i,j} \alpha_i \alpha_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle - \sum_i \alpha_i \left\| \mathbf{x}_i \right\|^2 \right) \\ \text{s.t.} \quad & \sum_i \alpha_i = 1, \ \forall i \ 0 \le \alpha_i \le \gamma, \end{aligned} \tag{5}$$

where $\langle - \rangle$ represents the (possibly kernelized) inner product. The α_i are Lagrange multipliers and $\gamma \in [1/n, 1]$ is a ceiling parameter that can be set to a value less than one to reduce the influence of outliers. The objective function is convex so a global minimum exists. In the kernelized case, the dual formulation yields a sparse solution in terms of the support vectors (the examples lying exactly on the hypersphere), which makes evaluating the model more efficient.

If we are given negative training samples, they can be used to improve the model by forcing them to lie outside of the bounding hypersphere. Suppose that we have n_1 training samples from the positive (object) class enumerated by indices i, j, and n_2 from the negative (background) class enumerated by l, m. The most compact hypersphere that includes the positive samples and excludes the negative ones can be found by solving the quadratic programming problem

$$\begin{aligned} \arg\min_{\mathbf{c},\, r \ge 0,\, \boldsymbol{\xi} \ge 0} \quad & \left(r^2 + \gamma_1 \sum_i \xi_i + \gamma_2 \sum_l \xi_l \right) \\ \text{s.t.} \quad & \left\| \mathbf{x}_i - \mathbf{c} \right\|^2 \le r^2 + \xi_i, \quad i = 1, \ldots, n_1 \\ & \left\| \mathbf{x}_l - \mathbf{c} \right\|^2 \ge r^2 - \xi_l, \quad l = 1, \ldots, n_2 \end{aligned} \tag{6}$$

or its dual

$$\begin{aligned} \arg\min_{\boldsymbol{\alpha}} \quad & \sum_{i,j} \alpha_i \, \alpha_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle + \sum_{l,m} \alpha_l \, \alpha_m \left\langle \mathbf{x}_l, \mathbf{x}_m \right\rangle \\ & - 2 \sum_{l,j} \alpha_l \, \alpha_j \left\langle \mathbf{x}_l, \mathbf{x}_j \right\rangle + \left(\sum_l \alpha_l \left\| \mathbf{x}_l \right\|^2 - \sum_i \alpha_i \left\| \mathbf{x}_i \right\|^2 \right) \\ \text{s.t.} \quad & \sum_i \alpha_i - \sum_l \alpha_l = 1, \ \forall i, j \ 0 \le \alpha_i \le \gamma_1, \ 0 \le \alpha_l \le \gamma_2. \end{aligned} \tag{7}$$

This one-class model differs from a classical SVM in that it finds a closed hypersphere surrounding the object class, not a linear hyperplane separating it from the background. We find that the inclusion of negative training samples significantly improves the performance of our detection cascades – particularly in cases where there are relatively few positive training examples, as in the person detector below – so we use the formulation (7) above.

Like (5), (7) is a quadratic program with a global minimum. Large-scale problems can be solved using Sequential Minimal Optimization (SMO) [20]. In particular, it is not necessary to construct the full Hessian matrix: only the Hessians of the active sets of samples need to be considered in each iteration. We revised the CMP quadratic programming software[2] to allow us to solve problems with millions of variables in a reasonable time. Given the optimal multipliers α, the center of the bounding hypersphere can be computed as

$$\mathbf{c} \ = \ \textstyle\sum_i \alpha_i \, \mathbf{x}_i - \sum_l \alpha_l \, \mathbf{x}_l \, , \tag{8}$$

after which its radius can be found using the constraints from (6).

During object detection, we find the distance from the feature vector of each local window to the center of the hypersphere, rejecting the sample as a negative if this distance is greater than the radius. This can be done very efficiently as it only requires vector subtraction and norm.

3.3. Nonlinear One-Class Classifier

The third stage of our cascade contains a single kernelized hypersphere classifier that makes the final decisions. Kernelization allows finer discrimination than the preceding linear stages in return for increased computation for the few examples that reach this stage. The hypersphere model can be kernelized simply by replacing the inner products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ with kernel evaluations $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ in (7), where $\phi()$ is the implicit feature space mapping implemented by the kernel. Training remains straightforward, but evaluating distances from incoming samples $\mathbf{x}$ to the center of the bounding hypersphere requires kernel evaluations $k(\mathbf{x}_i, \mathbf{x})$ against the support vectors $\mathbf{x}_i$. This makes kernelized SVDD significantly more expensive than its linear counterpart as the number of support vectors can be considerable (albeit typically much smaller than the number of training samples). However in practice we find that our kernelized SVDD classifiers are an order of magnitude faster than the analogous kernelized SVM's because they have far fewer support vectors. The SVM's typically have many negative support vectors owing to the need to reject large numbers of hard negatives, whereas the SVDD support vectors come predominantly from the positive training samples. This makes kernel SVDD more suitable for use in efficient detection cascades than kernel SVM.

4. Face Detection Experiments

We will evaluate our approach[3] on face detection and human detection tasks. First consider face detection. We tested on two datasets, the 13127 image "Faces in the Wild" one [11] and ESOGU[4], a new frontal face detection dataset that includes 285 high-resolution color images with 970 annotated frontal faces. The images in Faces in the Wild

[2] http://cmp.felk.cvut.cz

[3] For our code, see http://www2.ogu.edu.tr/∼mlcv/softwares.html

[4] http://mlcvdb.ogu.edu.tr/facedetection.html

Faces in the Wild	LBP+HOG			LTP+HOG			LBP+LTP			LBP+LTP+HOG		
Method	**DR**	**FP**	**AP**	**DR**	**FP**	**AP**	**DR**	**FP**	**AP**	**DR**	**FP**	**AP**
Cascade I	94.85	2852	95.73	95.81	3217	96.53	83.93	4281	90.34	88.64	837	97.81
Cascade II	96.60	332	98.58	95.84	254	98.62	88.49	920	95.95	95.10	234	98.60
Cascade III	98.36	237	98.80	**98.58**	**265**	**98.91**	93.24	849	98.04	97.20	626	98.67

ESOGU Faces	LBP+HOG			LTP+HOG			LBP+LTP			LBP+LTP+HOG		
Method	**DR**	**FP**	**AP**	**DR**	**FP**	**AP**	**DR**	**FP**	**AP**	**DR**	**FP**	**AP**
Cascade I	91.24	550	95.67	92.37	406	96.27	87.42	659	91.18	87.20	207	96.37
Cascade II	**92.47**	**18**	**99.01**	91.75	38	98.69	89.28	222	93.05	92.68	35	98.38
Cascade III	92.27	10	98.67	**94.02**	**71**	**98.72**	90.31	136	92.53	94.02	166	97.94

Table 1. % Detection Rates (DR), numbers of False Positives (FP) and % Average Precision (AP) scores for our cascade detectors on the Faces in the Wild and ESOGU Faces datasets. The 'Cascade I' detectors include only the linear hyperplane and hypersphere stages. The 'Cascade II' and 'Cascade III' ones respectively add a kernelized hypersphere classifier and a kernelized SVM as the final stage. For comparison, the OpenCV Viola-Jones detector [27] has DR 95.80%, FP 1074 and AP 98.50% on Faces in the Wild and DR 75.36%, FP 103 and AP 98.60% on ESOGU, and the FDLib detector [14] has DR 59.28% and FP 5393 on Faces in the Wild and DR 63.81% and FP 344 on ESOGU. (We can not report AP scores for the FDLib detector as it does not return a real-valued confidence measure for its detections). Note that the Faces in the Wild results are probably biased towards Viola-Jones because a detector of this kind was used to obtain the initial detections for this dataset [11].

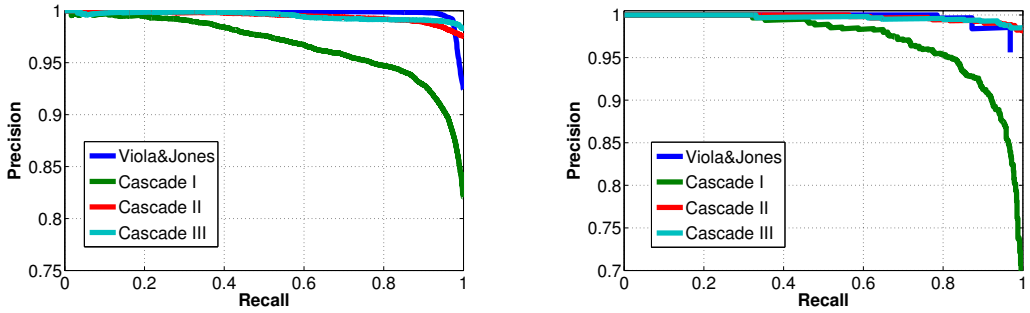

Figure 2. Precision-Recall curves for LBP+HOG features on the Faces in the Wild (left) and ESOGU Faces (right) datasets.

are somewhat idealized in the sense that they are relatively small and normalized such that most of the faces appear near the middle of the image with similar scales. To provide more realistic testing on images from real world consumer snapshot collections, we therefore developed the ESOGU (ESkisehir OsmanGazi University) dataset, whose images contain faces appearing at a wide range of image positions and scales, and also complex backgrounds, occlusions and illumination variations – *c.f.* Fig. 3 bottom.

Training: Given the limitations of the current publicly available face detector training datasets, we collected 12 500 subimages of frontal upright faces from the web for training. Most of these are from real-world images and there is a high degree of variability in appearance and lighting conditions. The face images are rescaled and aligned to a resolution of 35×28 (further reductions in resolution reduce the performance). For the negative set, we randomly selected 10 000 windows from face-free regions with complex backgrounds. We tested several visual features including LBP [1], LTP [24], HOG [6], and combinations of these. The combinations gave better results than the individual descriptors. For LBP and LTP, we divided the images into four non-overlapping quadrants and extracted descriptors from each region using circular (8,1) neighborhoods. The resulting histograms were normalized to sum 1 and concatenated to produce the final feature vector. For HOG, we used a grid of 6×6 pixel cells with 9 bins of unsigned gradient orientation over color images, grouping each cell into overlapping 2×2 cell blocks for normalization as in [9].

Classifiers trained with the initial samples were used to scan a set of thousands of images in order to collect both false negatives and false positives. These hard examples were added to the training set, increasing the number of positive examples to about 20k and the number of negative ones to about 93k, and the methods were retrained. The final size of the training set is thus 113k. When scanning an image the detection window is stepped by 3 pixels horizontally and 4 pixels vertically, and we scan an image pyramid whose scales are spaced by a factor of 1.15. For nonmaximum suppression we sort the surviving windows by score, then iteratively take the first and eliminate all detections

Figure 3. Some examples of the output of our cascade detectors on images from the Faces in the Wild (top) and ESOGU Faces (bottom) datasets. Most of the faces are correctly detected, but there are a few missed detections and false positives.

overlapping it. To penalize narrow supports, groups with less than 4 overlapping windows (8 in the linear case) are suppressed, and otherwise $\log(\#\text{ participating windows})/3$ is heuristically added to the score.

Detectors: We trained three kinds of cascades. The first includes only the linear hyperplane and linear hypersphere stages, with three linear hyperplane classifiers in the first stage. The second and third cascades are three-stage ones, respectively with kernelized one-class hypersphere classifiers and kernelized SVM's in the final stage. For these, only two linear hyperplane classifiers were included in the first stage. We used Gaussian RBF kernels as they produced better results than RBF kernels based on the χ^2 histogram distance. The kernelized hypersphere classifiers always had fewer support vectors than the corresponding kernelized SVM's. For example for LBP+HOG features, the hypersphere classifier had 2398 support vectors while the SVM had 15 657 – 6.5 times more. On average, the final stages of cascades using kernelized hypersphere classifiers were 8 times faster than ones using kernelized SVM's.

We compared our results with those of the OpenCV Viola-Jones cascade [27] and the FDLib detector [14][5], using the PASCAL VOC criteria [7] to assess detection per-

[5]http://people.kyb.tuebingen.mpg.de/kienzle/fdlib/fdlib.htm

formance. Briefly, detections are considered to be true positives if the bounding box R returned by the classifier overlaps the bounding box Q of the ground truth annotation by more than 50%, where overlap is measured as $\frac{\text{Area}|Q \cap R|}{\text{Area}|Q \cup R|}$. We report the Detection Rate (DR) and total number of False Positives (FP) at the default detector threshold (the one chosen by the training algorithm), as well as the Average Precision (AP) (*i.e.* area under curve) over the whole Precision-Recall curve. DR is the ratio of the number of correctly detected faces to the total number of labeled faces in the test set.

Results: The results are given in Table 1 and Fig. 2, and Fig. 3 shows some examples of detections on the two face test sets. For Faces in the Wild, cascades using final kernel SVM's have slightly higher AP's than ones using final kernel hypersphere classifiers, but this is reversed for ESOGU and in any case the differences are very small. For both datasets the Viola-Jones method comes a close third to the two cascades, while the FDLib detector gives poor results. The best feature set for Faces in the Wild is LTP+HOG whereas the best for ESOGU is LBP+HOG, but for both datasets the feature combinations LBP+HOG, LTP+HOG and LBP+LTP+HOG all give similar results, with LBP+LTP and the individual feature sets (not shown) being weaker. This suggests that HOG manages to capture useful cues (probably shape information) that LBP and LTP ignore, and conversely LBP and LTP capture cues (probably local texture) that HOG ignores. Given that there is no clear winner among LBP+HOG, LTP+HOG and LBP+LTP+HOG, we recommend LBP+HOG for this application as it has lower computational complexity than the other combinations. To get an idea of the degree of pruning provided by the cascades, for LBP+HOG features on the ESOGU dataset, of the 23M windows scanned, 2.9M (13%) passed the first hyperplane classifier, 0.8M (3.6%) passed the second hyperplane, 64k (0.28%) passed the linear hypersphere stage and 21k (0.09%) passed the kernel hypersphere stage. Nonmaximum suppression then merged these into 915 detections (an average of 22.7 windows per detection), of which 897 were correct.

5. Human Detection Experiments

Training: We used the INRIA Person dataset [6] for our human ("pedestrian") detection experiments. LBP+HOG features were used, with a grid of 8×8 pixel cells for HOG and the detection window divided into a 5×3 set of rectangular regions for LBP. We artificially enlarged the positive training set by slightly perturbing the locations provided in the ground-truth annotations, and randomly sampled 12180 negative windows from the provided negative (person-free) training images. For each method tested, initial detectors trained on these examples were used to scan all of the training images to collect hard examples, followed by retraining. During detection, the search window was shifted by steps of 4 pixels horizontally and 6 pixels vertically, and the pyramid scales were spaced by a factor of 1.15. We trained the same three kinds of cascades as in the face case. The final kernel hypersphere classifier had 2818 support vectors whereas the final kernel SVM had 10 times more (28 251).

Method	Det.Rate	False Pos.	Ave.Prec.
Cascade I	78.19	497	90.43
Cascade II	75.67	144	93.46
Cascade III	82.83	104	96.03
Dalal&Triggs [6]	-	-	75.00
Hussain&Triggs [12]	-	-	84.10
Felzenszwalb et al. [9]	-	-	86.90

Table 2. % Detection Rates, total numbers of False Positives, and % Average Precision scores) for our detectors on the INRIA Person dataset.

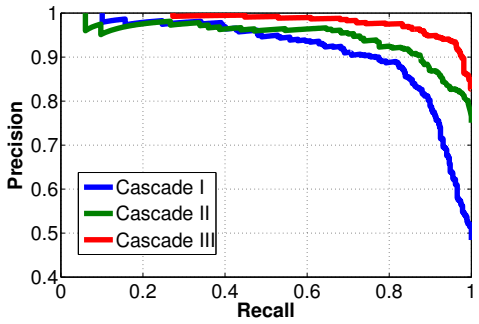

Figure 4. Precision-Recall curves for our detectors on the INRIA Person dataset.

Results: The Detection Rates and Average Precision scores for our person detectors are given in Table 2 and the corresponding Precision-Recall curves are given in Figure 4. For comparison, we also include the published AP's of Dalal & Triggs [6] (HOG with linear SVM), Hussain & Triggs [12] (a two stage linear + quadratic single root latent SVM classifier using HOG+LBP+LTP) and Felzenszwalb *et al.* [9,8] (a linear latent SVM classifier using multiple roots and parts over HOG). All of our cascades give better results than these methods, despite the fact that they use only a single root and no parts. The cascade with the final kernel SVM clearly dominates, giving the best Detection Rate, False Positives and AP scores and offering about a 9% improvement in AP over the previous state-of-art. The cascade with the final kernel hypersphere classifier comes second, with its final stage being about 20 times faster than that of the kernel SVM based one. The two-stage cascade based on linear classifiers also achieves very respectable results, which suggests that our strategy of bounding the region occupied by the positive class more tightly than the simple linear separator provided by an SVM is bearing fruit.

6. Summary and Conclusions

We have developed sliding window object detectors based on short cascades of linear and nonlinear nearest-convex-model classifiers, arguing that the "one-class" nature of the latter provides an attractive combination of accuracy and speed. Our cascades have three stages: a set of linear distance-to-hyperplane classifiers for fast pruning of easy negatives; a linear hypersphere classifier for additional pruning; and finally (and optionally) either a kernelized hypersphere classifier or a kernelized SVM. We tested our detectors on two challenging face datasets and the INRIA Person dataset, concluding that the cascade methods are very promising relative to existing approaches. In particular, the cascades with final kernelized classifiers achieve high Average Precisions, with the hypersphere ones having accuracy similar to or better than the SVM ones on the face datasets and somewhat lower on the INRIA dataset, but being an order of magnitude faster in both cases because they have far fewer support vectors. For human detection, the two-stage linear cascade already gives much better performance than well-established linear SVM detectors, which suggests that including multiple stages of linear or hypersphere pruning may be a useful strategy for improving other existing object detectors.

Future work: Unlike [8], our current detectors do not incorporate multiple roots, parts, and latent position and scale adjustments during training (although the use of perturbed training examples partially compensates for the latter). We are currently working on including these refinements.

Acknowledgments: We would like to thank Jifeng Shen for supplying some of the training images. This work was funded in part by the Scientific and Technological Research Council of Turkey (TUBİTAK) under Grant number EEEAG-109E279.

References

[1] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE T-PAMI*, 28(12):2037–2041, 2006.

[2] D. Aldavert, A. Ramisa, R. L. Mantaras, and R. Toledo. Fast and robust object segmentation with the integral linear classifier. In *CVPR*, 2010.

[3] Y. Amit and D. Geman. A computational model for visual selection. *Neural computation*, 11:1691–1715, 1999.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *CVIU*, 110(3):346–359, 2008.

[5] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE T-PAMI*, 24(24):509–521, 2002.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[7] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge. *IJCV*, 88(2):303–338, 2010.

[8] P. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE T-PAMI*, 32(9), Sept. 2010.

[9] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale deformable part model. In *CVPR*, 2008.

[10] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *ICCV*, 2009.

[11] G. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, Oct. 2007.

[12] S. Hussain and B. Triggs. Feature sets and dimensionality reduction for visual object detection. In *BMVC*, 2010.

[13] H. Jin, Q. Liu, and H. Lu. Face detection using one-class-based support vectors. In *International Conference on Automatic Face and Gesture Recognition*, 2004.

[14] W. Kienzle, G. Bakir, M. Franz, and B. Scholkopf. Face detection – efficient and rank deficient. In *NIPS*, pages 673–680, 2005.

[15] K. Levi and Y. Weiss. Learning object detection from a small number of examples: the importance of good features. In *CVPR*, 2004.

[16] D. G. Lowe. Distinctive image features from scale invariant keypoints. *IJCV*, 60:91–110, 2004.

[17] O. L. Mangasarian and E. W. Wild. Multisurface proximal support vector machine classification via generalized eigenvalues. *IEEE T-PAMI*, 28:69–74, 2006.

[18] K. Mele and J. Maver. Object recognition using hierarchical SVMs. In *Computer Vision Winter Workshop*, 2003.

[19] C. Papageorgiou and T. Poggio. A trainable system for object detection. *IJCV*, 38:15–33, 2000.

[20] J. C. Platt. Fast training of support vector machines using sequential minimal optimization, 1998. Advances in Kernel Methods-Support Vector Learning, Cambridge, MA, MIT Press.

[21] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE T-PAMI*, 20:22–38, 1998.

[22] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE T-PAMI*, 20:23–38, 1998.

[23] L. Shams and J. Speslstra. Learning Gabor-based features for face detection. In *World Congress in Neural Networks*, 1996.

[24] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Transactions on Image Processing*, 19:1635–1650, 2010.

[25] D. M. J. Tax and R. P. W. Duin. Support vector data description. *Machine Learning*, 54:45–66, 2004.

[26] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.

[27] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[28] X. Wang, T. X. Han, and S. Yan. A HOG-LBP human detector with partial occlusion handling. In *ICCV*, 2009.