PERGAMON

Pattern Recognition 36 (2003) 2007–2017

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Hierarchical classification and feature reduction for fast face detection with support vector machines

Bernd Heisele[a,b,*], Thomas Serre[b], Sam Prentice[b], Tomaso Poggio[b]

[a] *Honda Research Institute US, 145 Tremont St., Boston, MA 02111, USA*
[b] *Center for Biological and Computational Learning, MIT, E25-201, 45 Carleton St., Cambridge, MA 02142, USA*

Accepted 15 January 2003

Abstract

We present a two-step method to speed-up object detection systems in computer vision that use support vector machines as classifiers. In the first step we build a hierarchy of classifiers. On the bottom level, a simple and fast linear classifier analyzes the whole image and rejects large parts of the background. On the top level, a slower but more accurate classifier performs the final detection. We propose a new method for automatically building and training a hierarchy of classifiers. In the second step we apply feature reduction to the top level classifier by choosing relevant image features according to a measure derived from statistical learning theory. Experiments with a face detection system show that combining feature reduction with hierarchical classification leads to a speed-up by a factor of 335 with similar classification performance.

Keywords: Face detection; Object detection; Feature reduction; Hierarchical classification; Support vector machines

1. Introduction

A major task in visual scene analysis is to detect objects in images. This is commonly done by shifting a search window over an input image and by categorizing the object in the window with a classifier. The main problem in categorization is the large range of possible variations within a class of objects. The system must generalize not only across different viewing and illumination conditions but also across different exemplars of a class, such as faces of different people for face detection. This requires complex, computationally expensive classifiers. Further contributing to the computational load of the task is the large amount of input data that has to be processed. A real-time vision system has to deal with data streams in the range of several MBytes/s. Speeding-up classification is therefore of major concern when developing systems for real-world applications. In the following we investigate two methods for speed-ups: hierarchical classification and feature reduction.

In Ref. [1] we presented a system for detecting frontal and near-frontal views of faces in still gray images. The system achieved high detection accuracy by classifying 19×19 gray patterns using a non-linear SVM. Searching an image for faces at different scales took several minutes on a PC, far too long for most real-world applications. Experiments with faster classifiers (linear SVMs) gave significantly lower recognition rates. To speed-up the system without losing in classification performance one can exploit the following two characteristics common to most vision-based detection tasks: First, the vast majority of the analyzed patterns in an image belongs to the background class. For example, the ratio of non-face to face patterns in the tests in Ref. [1] was about 50,000 to 1. Second, many of the background patterns can be easily distinguished from the objects. Based on these

* Corresponding author. Honda Research Institute US, 145 Tremont St., Boston, MA 02111, USA. Tel.: +1-617-338-4909; fax: +1-617-338-4085.
E-mail addresses: heisele@ai.mit.edu (B. Heisele), serre@ai.mit.edu (T. Serre), prentice@mit.edu (S. Prentice), tp@ai.mit.edu (T. Poggio).

0031-3203/03/$30.00
doi:10.1016/S0031-3203(03)00062-1

two task-specific characteristics it is sensible to apply a hierarchy of classifiers. Fast classifiers remove large parts of the background on the bottom and middle levels of the hierarchy and a more accurate but slower classifier performs the final detection on the top level. This idea is related to the well known coarse-to-fine template matching [2–4]. In Ref. [5] hierarchical classification is used to speed-up a face detection system. A candidate selection neural network with increased robustness against translation is added as a first layer to an existing face detection neural network.

We present a method for building and training a hierarchy of SVM classifiers given a set of classifiers which operate at different image resolutions. The iterative algorithm starts with the topmost classifier at the highest image resolution and adds a new lower layer. At each iteration, the hierarchies are retrained bottom up and a speed test is performed on a validation set of non-face images to choose the hierarchy with the least number of computations.

Another possibility of speeding-up classification is to reduce the number of features by selecting a subset of relevant features. Feature reduction is a more generic tool than the above described hierarchical classification and can be applied to any classification problem. There are basically two types of feature selection methods in the literature: filter and wrapper methods [6]. Filter methods are preprocessing steps performed independently of the classification algorithm or its error criteria. Wrapper methods attempt to search through the space of feature subsets using the criterion of the classification algorithm to select the optimal feature subset [7]. We present a new wrapper method to reduce the dimensions of both input and feature space of a non-linear SVM. For our final detection system we combine feature selection with hierarchical classification by putting a non-linear SVM with feature selection on top of the hierarchy of linear SVMs. A similar idea of combining feature selection with hierarchical classification has been recently proposed in Ref. [8] for frontal face detection. They use AdaBoost to train a cascade of linear classifiers and to select features from an over complete set of Haar wavelet features. In contrast to our approach, however, the complexity of the classifiers in the final hierarchy is only controlled by the number of features and not by the class of decision functions (i.e. class of kernel functions).

The outline of the paper is as follows: In Section 2 we give a brief overview of SVM classification. In Section 3 we describe how to build and train the hierarchical system. Sections 4 and 5 describe the feature selection methods for the input and the feature space of the SVM, respectively. In Section 6 we present experimental results of the hierarchical system with feature reduction. The paper is concluded in Section 7.

2. Background on support vector machines

2.1. Theory

An SVM [9] performs pattern recognition for a two-class problem by determining the separating hyperplane that has maximum distance to the closest points of the training set. These closest points are called support vectors. To perform a non-linear separation in the input space a non-linear transformation $\Phi(\cdot)$ maps the data points $\mathbf{x}$ of the input space $\mathbb{R}^n$ into a high-dimensional space, called feature space $\mathbb{R}^p$ ($p > n$). The mapping $\Phi(\cdot)$ is represented in the SVM classifier by a kernel function $K(\cdot,\cdot)$ which defines an inner product in $\mathbb{R}^p$. Given ℓ examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{\ell}$, the decision function of the SVM is linear in the feature space and can be written as

$$f(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}) + b = \sum_{i=1}^{\ell} \alpha_i^0 y_i K(\mathbf{x}_i, \mathbf{x}) + b. \tag{1}$$

The optimal hyperplane is the one with the maximal distance in space $\mathbb{R}^p$ to the closest points $\Phi(\mathbf{x}_i)$ of the training data. Determining that hyperplane leads to maximizing the following functional with respect to α:

$$W^2(\alpha) = 2\sum_{i=1}^{\ell} \alpha_i - \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2}$$

under constraints $\sum_{i=1}^{\ell} \alpha_i y_i = 0$ and $C \geqslant \alpha_i \geqslant 0$, $i = 1, \ldots, \ell$. An upper bound on the expected error probability EP_{err} of an SVM classifier is given by [9]

$$EP_{err} \leqslant \frac{1}{\ell} E\left(\frac{R^2}{M^2}\right) = \frac{1}{\ell} E(R^2 W^2(\alpha^0)), \tag{3}$$

where $M = 1/W(\alpha^0)$ is the distance between the support vectors and the separating hyperplane and R is the radius of the smallest sphere including all points $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_\ell)$ of the training data in the feature space. In the following, we will use this bound on the expected error probability to rank and select features.

2.2. Computational issues

The only non-linear kernel investigated in this paper is a second-degree polynomial kernel $K(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x} \cdot \mathbf{y})^2$ which has been successfully applied to various object detection tasks [1,10]. Eq. (1) shows two ways of computing the decision function. When using the kernel representation on the right side of Eq. (1) the number of multiplications required to calculate the decision function for a second-degree polynomial kernel is

$$M_{k,poly2} = (n + 2) \cdot s, \tag{4}$$

where n is the dimension of the input space and s is the number of support vectors. This number is independent of the dimensionality of the feature space. It depends on the number of support vectors which is linear with the size ℓ of the training data [9]. On the other hand, the computation of the decision function in the feature space is independent of the size of training samples, it only depends on the dimensionality p of the feature space. For the second-degree polynomial kernel the feature space $\mathbb{R}^p$ has dimension $p = (n + 3)n/2$ and is given by $\mathbf{x}^* = (\sqrt{2}x_1, \ldots, \sqrt{2}x_n, x_1^2, \ldots, x_n^2, \sqrt{2}x_1x_2, \ldots, \sqrt{2}x_{n-1}x_n)$.

Thus the number of multiplications required for projecting the input vector into the feature space and for computing the decision function is

$$M_{f,poly2} = \frac{(n+1)n}{2} + \frac{(n+3)n}{2} = (n+2)\cdot n. \quad (5)$$

From Eqs. (4) and (5) we see that the computation for an SVM with second-degree polynomial is more efficiently done in the feature space if the number of support vectors is bigger than n. This was always the case in our experiments; the number of support vectors was between 2 and 6 times larger than n. That is why we investigated not only methods for reducing the number of input features but also methods for feature reduction in the feature space.

3. Hierarchy of classifiers

3.1. System overview

In most object detection problems the majority of analyzed image patches belong to the background class. Only a small percentage of them look similar to objects and require a highly accurate classifier to avoid false classifications. It is sensible to apply a hierarchy of classifiers where the complexity of the classifier increases with each level. By propagating only those patterns that were not classified as background, we quickly decrease the amount of data to process. The main issues in designing such a classification hierarchy are how to choose the input features to the classifiers, how to select the number of levels, and finally how to train the classifiers. We used pixel values as inputs to the classifiers and reduced the number of features from top to bottom by decreasing the image resolution, similar to coarse-to-fine matching approaches. An example of a hierarchical system with five layers is shown in Fig. 1.

3.2. Building the hierarchy

In the following we describe an algorithm for automatically determining the architecture of the hierarchy. We begin with a set of classifiers which operate at different resolutions and are each trained over the entire training set. In our experiments the resolution of the linear SVM classifiers ranged from 3×3 to 19×19. Given this set, the goal is to find the best hierarchy with respect to speed and recognition performance. The algorithm builds the hierarchy in an iterative top-down fashion starting with the topmost classifier and adding a new layer at each iteration. It consists of three steps:

(a) *Adding a new layer*. We add a classifier to the hierarchy which operates at a lower resolution than the current bottom classifier. For example, if the current hierarchy consists of a 19×19 and an 11×11 classifier we add classifiers

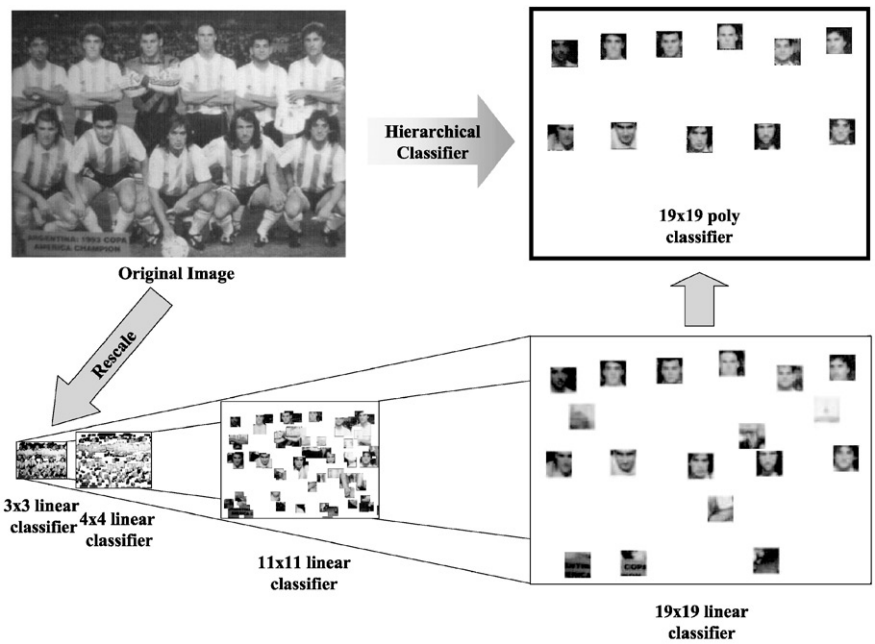

Fig. 1. Example of a hierarchical system with five levels: Starting with the classifier at the lowest resolution patterns which have been classified as background are successively removed from the image going up the hierarchy. The final non-linear classifier processes the image at maximum resolution.

Table 1
Number of negative training examples used for training in bottom-up training procedures

	3 × 3		4 × 4		11 × 11		19 × 19
Bottom-up:	33,045	→	23,598	→	18,149	→	10,568

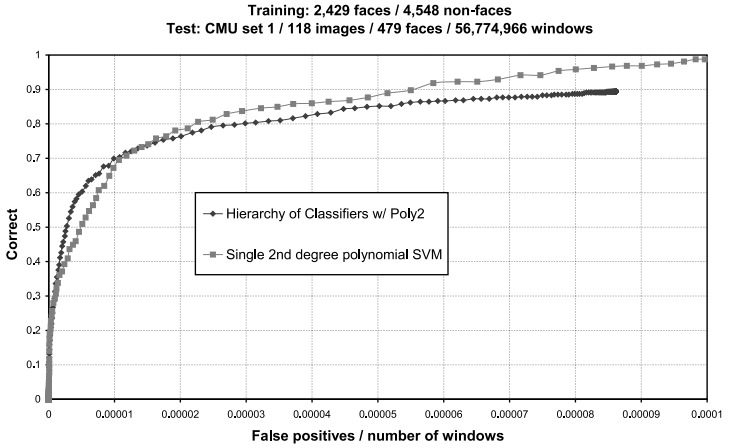

Fig. 2. ROC curves of the hierarchical system and the original system with a single second-degree polynomial SVM for the CMU set 1.

with resolutions ranging from 9 × 9 down to 3 × 3, resulting in 7 new hierarchies.

(b) *Retraining*. We perform bottom-up training of the new hierarchies. We train the bottom classifier on the whole training set and then shift its hyperplane such that a fixed percentage of positive examples [1] is on one side. Then we remove all negative examples which lie on the other side of the hyperplane to generate a new training set for the classifier on the next higher level. This is continued for each layer till we reach the top of the hierarchy.

(c) *Speed test*. We set the thresholds of all classifiers in the hierarchies such that a fixed percentage of the faces in the training set [2] is correctly classified. Then we perform a speed test of the hierarchies on the validation set of non-faces and choose the hierarchy with the least number of computations.

The iterative process is continued until the lowest resolution is reached or adding a new layer does not increase the speed of the system.

In our experiments we used a set of linear SVM classifiers with resolutions ranging from 3 × 3 to 19 × 19 and a non-linear SVM at resolution 19 × 19 which was chosen to be our top level classifier. The non-linear SVM was not included into the bottom-up learning procedure described above. For a realtime system this computationally expensive classifier can only be applied to a very small percentage of the input patterns. Since our bottom-up training method reflects the runtime data flow, the few training examples that remain would not have been sufficient to train an accurate classifier. The training set for the linear classifiers consisted of 9662 face images and 33,045 non-face images at resolution 19 × 19. The validation set consisted of 73,089 randomly collected non-face images at resolution 19 × 19. We applied histogram equalization as proposed in Ref. [11] to decrease the variations caused by changes in illumination. To train the classifiers at lower resolutions we downscaled the training images to the proper size. Applying the above described training procedure resulted in a five level hierarchy with the 19 × 19 non-linear SVM on top, followed by a 19 × 19, 11 × 11, 4 × 4, and finally 3 × 3 linear SVM classifier. The number of examples removed in the bottom-up training procedure for the final hierarchy is given in Table 1.

As mentioned before, the topmost classifier was trained independently. We used the same training set (2429 faces and 4548 non-face patterns) as in Ref. [1] to allow for comparisons with the results reported there. In Fig. 2 we show the ROC curves of the single second-degree polynomial classifier [1] and our hierarchical system on the CMU set 1 [3]. We selected the thresholds of the hierarchical classifiers in layers one to four according to the individual ROC

[1] Was set to 99% in the experiments.

[2] Was set to 99% in the experiments.

[3] This set [12] consists of 118 images including 479 faces. In our experiments searching over multiple scales and locations resulted in processing of about 57,000,000 19 × 19 patterns.

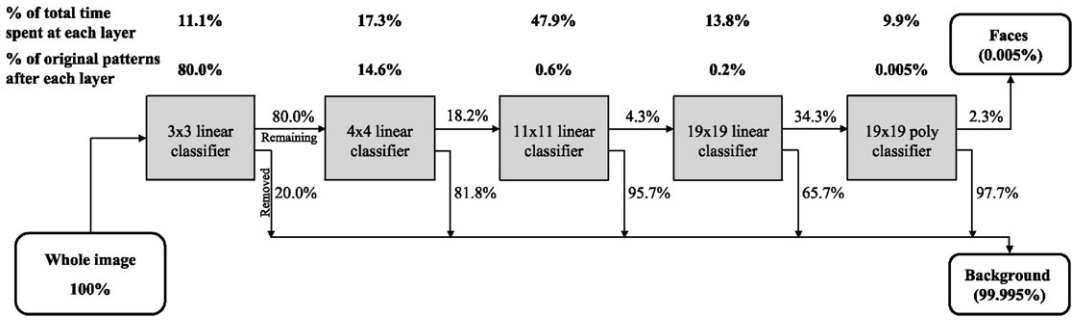

Fig. 3. Data flow and computing time for the hierarchical classifier tested on the CMU set 1.

curves on the training set[4]. The hierarchy performs better than the single classifier for recognition rates below 75%. Above that, the single SVM classifier is superior, indicating that some of the difficult face patterns in the test set do not reach the last layer of the hierarchy. Fig. 3 shows the data flow through the hierarchy and the time spent in each layer. The hierarchical system is about 260 times faster than the system with the single second-degree polynomial SVM.

In layers one to four most of the computation time ($\sim 90\%$) is used for feature extraction. Further optimizing the classifiers would not lead to a significantly faster system. In the last layer about 95% of the computing time is spent on the classification leaving much room for speed-ups using feature reduction methods. In the following sections we explore methods for feature reduction and apply them to the non-linear SVM at the top level of the hierarchy.

4. Dimension reduction in the input space

4.1. Ranking features in the input space

In Ref. [13] a gradient descent method is proposed to rank the input features by minimizing the bound of the expectation of the leave-one-out error of the classifier. The algorithm showed superior performance compared to other feature selection methods (filter methods based on Fisher score, Pearson correlation coefficients and Kolmogorov–Smirnov test) for various classification tasks (face detection, person detection, cancer morphology classification). The basic idea is to re-scale the n-dimensional input space by a $n \times n$ diagonal matrix σ such that R^2/M^2 is minimized. The new mapping function is then $\Phi_\sigma(\mathbf{x}) = \Phi(\sigma \cdot \mathbf{x})$ and the kernel function is $K_\sigma(\mathbf{x}, \mathbf{y}) = K(\sigma \cdot \mathbf{x}, \sigma \cdot \mathbf{y}) = (\Phi_\sigma(\mathbf{x}) \cdot \Phi_\sigma(\mathbf{y}))$. The decision function given in Eq. (1) becomes

$$f(\mathbf{x}, \sigma) = \mathbf{w} \cdot \Phi_\sigma(\mathbf{x}) + b = \sum_{i=1}^{\ell} \alpha_i^0 y_i K_\sigma(\mathbf{x}_i, \mathbf{x}) + b. \tag{6}$$

[4] We selected the point on the ROC curve with 97% recognition rate.

The maximization problem of Eq. (2) is now given by

$$W^2(\alpha, \sigma) = \max_{\alpha} \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j K_\sigma(\mathbf{x}_i, \mathbf{x}_j) \tag{7}$$

subject to constraints $\sum_{i=1}^{\ell} \alpha_i y_i = 0$ and $C \geqslant \alpha_i \geqslant 0,\ i = 1, \ldots, \ell$. The radius around the data is computed by solving the following maximization problem:

$$R^2(\beta, \sigma) = \max_{\beta} \sum_i \beta_i K_\sigma(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j} \beta_i \beta_j K_\sigma(\mathbf{x}_i, \mathbf{x}_j) \tag{8}$$

subject to $\sum_i \beta_i = 1,\ \beta_i \geqslant 0,\ i = 1, \ldots, \ell$.

We solve for α, β, and σ using an iterative procedure: We initialize σ as a vector of ones and then solve Eqs. (7) and (8) for the margin and radius, respectively. Using the values for α and β and the bound in Eq. (3) we compute σ by minimizing $W^2(\alpha, \sigma) R^2(\beta, \sigma)$ using a gradient descent procedure. We then start a new iteration of the algorithm using the σ of the current iteration as initialization.

We applied the ranking method to 283 gray features generated by preprocessing 19×19 image patterns as described in Ref. [11]. Additionally we performed tests with features obtained by projecting the data points into the 283 dimensional eigenvector space. The Principal Component Analysis (PCA) was computed on the combination of the positive and negative training sets. We computed one iteration of the gradient descent algorithm and performed tests for 60, 80 and 100 ranked features. The ROC curves for second-degree polynomial SVMs are shown in Fig. 4.

In experiments with 100 features there was no difference between gray and PCA features. For 80 and 60 features, however, the PCA gray features were superior. For this reason we work exclusively with PCA gray features in the following section.

4.2. Selecting features in the input space

In Section 4.1 we ranked the features according to their scaling factors σ_i. Now the problem is to determine a subset of the ranked features $(x_1, x_2, \ldots, x_n)$. This problem can be formulated as finding the optimal subset of ranked features $(x_1, x_2, \ldots, x_{n^*})$ among the n possible subsets, where

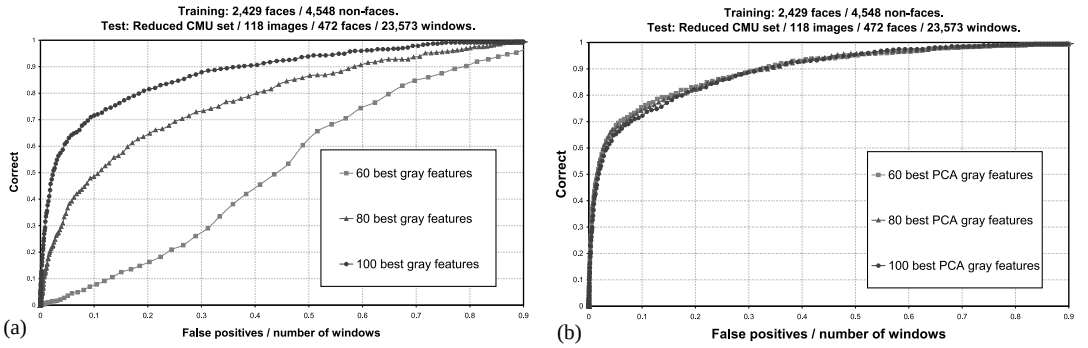

Fig. 4. ROC curves for (a) gray features and (b) PCA gray features with the 60, 80 and 100 ranked features.

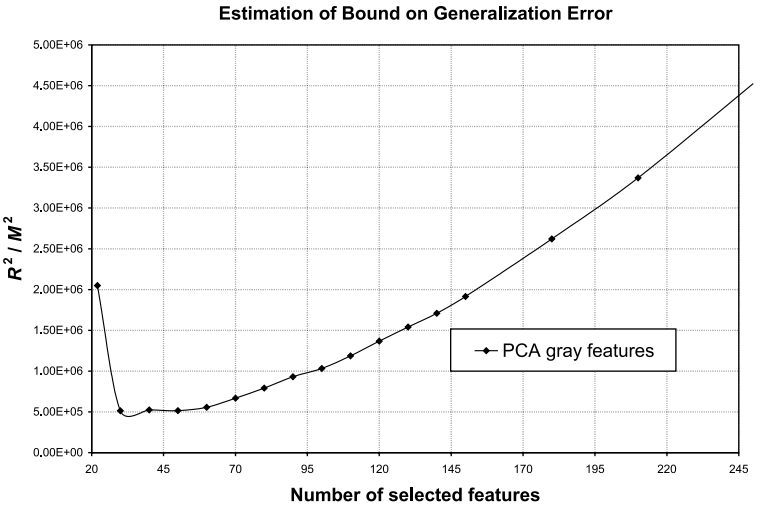

Fig. 5. Approximation of estimated bound on the expected error versus number of principal components. The values on the y-axis are not normalized by the number of training samples.

n^* is the number of selected features. As a measure of the classification performance of an SVM for a given subset of ranked features we again use the bound on the expected error probability:

$$EP_{err} \leqslant \frac{1}{\ell} E\left(\frac{R^2}{M^2}\right). \tag{9}$$

To simplify the computation of our algorithm and to avoid solving a quadratic optimization problem in order to compute the radius R, we approximated[5] R^2 by $2p$ where p is the dimension of the feature space $\mathbb{R}^p$. For a second-degree polynomial kernel of type $(1 + \mathbf{x} \cdot \mathbf{y})^2$ we get

$$EP_{err} \leqslant \frac{1}{\ell} 2pE(W^2(\alpha^0)) \leqslant \frac{1}{\ell} n^*(n^* + 3)E(W^2(\alpha^0)), \tag{10}$$

where n^* is the number of selected features[6]. The estimated bound on the expected error is shown in Fig. 5. We had no training error for more than 22 selected features. The bound drops over the first 30 features, stays about the same between 30 and 60 features, and then increases steadily. This bound is considered to be a loose bound on the expected error. To check if it is of practical use for selecting the number of features we performed tests on the CMU set 1. In Fig. 6 we compare the ROC curves obtained for different numbers of selected features. The results show that using more than 60 features does not improve the performance of the system. This observation coincides with the run of the curve in Fig. 5. However, the error on the test set does not change significantly for more than 70 features although

[5] We previously normalized all the data in $\mathbb{R}^n$ to be in a range between 0 and 1. As a result the points lay within a p-dimensional cube of length $\sqrt{2}$ in $\mathbb{R}^p$ and the smallest sphere including all the data points is upper bound by $\sqrt{2p}$.

[6] For a second-degree polynomial SVM the dimension of the feature space is $p = n^*(n^* + 3)/2$.

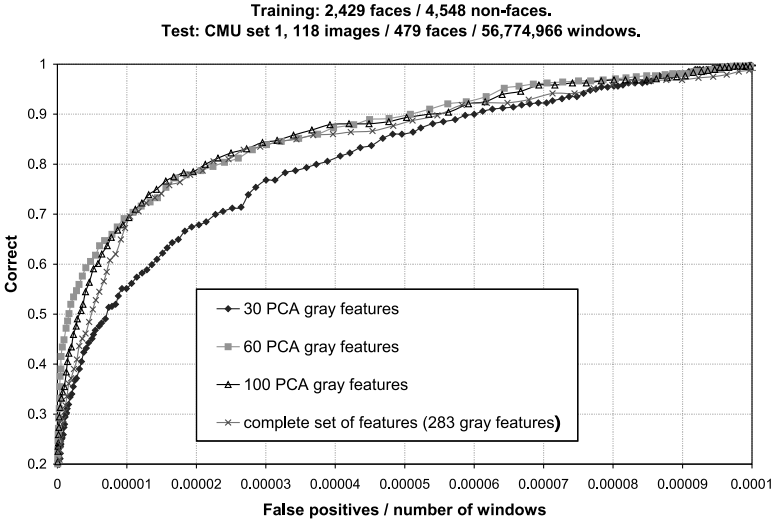

Fig. 6. ROC curves for different numbers of PCA gray features.

the estimated bound on the expected error shown in Fig. 5 increases. An explanation for this could be that the bound gets looser with increasing number of features because of our approximation of R by the dimensionality of the feature space.

5. Feature reduction in the feature space

In the previous Section we described how to reduce the number of features in the input space. Now we consider the problem of reducing the number of features in the feature space. We use a new method based on the contribution of the features from the feature space to the decision function $f(\mathbf{x})$ of the SVM.

$$f(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}) + b = \sum_{i=1}^{\ell} \alpha_i^0 y_i K(\mathbf{x}_i, \mathbf{x}) + b \tag{11}$$

with $\mathbf{w} = (w_1, \ldots, w_p)$. For a second-degree polynomial kernel with $K(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x} \cdot \mathbf{y})^2$, the feature space $\mathbb{R}^p$ with dimension $p = (n + 3)n/2$ is given by $\mathbf{x}^* = (\sqrt{2}x_1, \ldots, \sqrt{2}x_n, x_1^2, \ldots, x_n^2, \sqrt{2}x_1x_2, \ldots, \sqrt{2}x_{n-1}x_n)$. The contribution of a feature x_k^* to the decision function in Eq. (11) depends on w_k. A straightforward way to order the features is by ranking $|w_k|$. Alternatively, we weighted $\mathbf{w}$ by the support vectors to account for different distributions of the features in the training data. The features were ordered by ranking $|w_k \sum_i y_i x_{i,k}^*|$, where $x_{i,k}^*$ denotes the kth component of support vector i in feature space $\mathbb{R}^p$. For both methods we first trained an SVM with a second-degree polynomial kernel on 60 PCA gray features of the input space which corresponds to 1891 features in $\mathbb{R}^p$. We then calculated

$$E(S) = \frac{1}{s} \sum_i |f(\mathbf{x}_i) - f_S(\mathbf{x}_i)| \tag{12}$$

for all s Support Vectors, where $f_S(\mathbf{x})$ is the decision function using the S first features according to their ranking. Note that in contrast to the previously described selection of features in the input space this method does not require the retraining of SVMs for different feature sets. The results in Fig. 7 show that ranking by the weighted components of $\mathbf{w}$ lead to a faster convergence of $E(S)$ from Eq. (12) towards 0. Fig. 8 shows the ROC curves for 500 and 1000 features. As a reference we added the ROC curve for a second-degree SVM trained on the original 283 gray features. This corresponds to $(283 + 3)283/2 = 40,469$ components in the feature space. By combining both methods of feature reduction we could reduce the dimensionality by a factor of about 40 without loss in performance.

6. Experiments

In the final experiments we combined feature reduction with hierarchical classification. In Fig. 9 we compare the ROC curves of the hierarchical system and the single SVM classifier with second-degree polynomial kernel. The hierarchy performs better than the single classifier up to a recognition rate of 80%. For higher recognition rates, the single SVM classifier performs slightly better because some of the more difficult face patterns in the test set do not reach the last layer of the hierarchy. The average computing time on a Pentium IV with 1.8 GHz for an image of size 320×240 is given in Table 2. We speed-up the detection by a factor

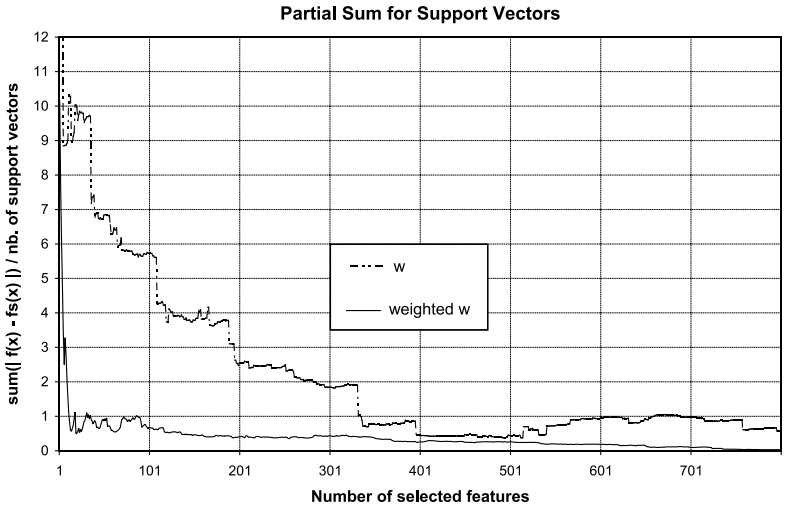

Fig. 7. Classifying support vectors with a reduced number of features. The x-axis shows the number of features, the y-axis is the mean absolute difference between the output of the SVM using all features and the same SVM using the S first features only. The features were ranked according to the components and the weighted components of the normal vector of the separating hyperplane.

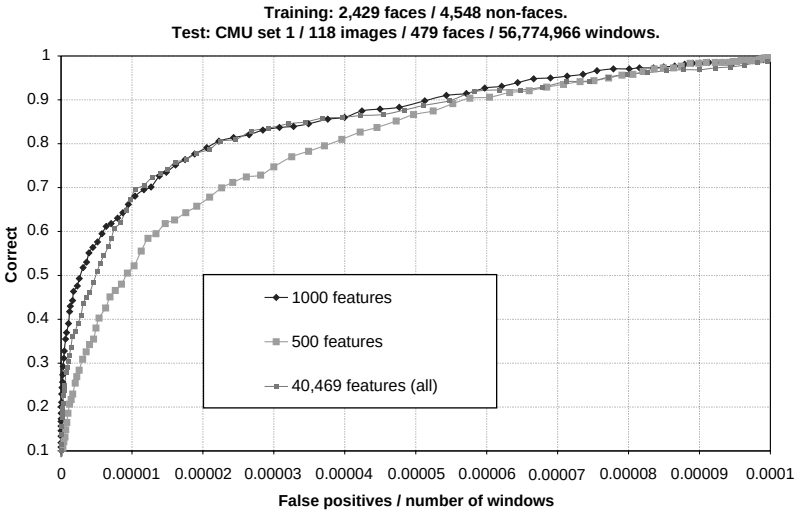

Fig. 8. ROC curves for different dimension of the feature space.

of 335 compared to the original system achieving a processing speed of 4 frames/s. The face detector proposed in Ref. [8] is about 4 times faster at runtime. However, it requires several weeks for training while our system can be trained in a day.

In Fig. 10 we show the results for an example image from CMU set 1. The original image is shown in Fig. 10(a), Fig. 10(b)–(e) show the outputs of the SVM classifiers at levels two to five, respectively. Dark pixels represent high output values of the SVM indicating the presence of a face, white areas have been removed by previous classifiers in the hierarchy. In Fig. 10(f) we show the final detection results as computed by the top level second-degree polynomial SVM classifier. Some more example images from CMU set 1 are shown in Fig 11.

7. Conclusion and future work

In this paper we presented speed-up methods for a face detection system based on hierarchical classification and feature reduction. To quickly remove large background parts

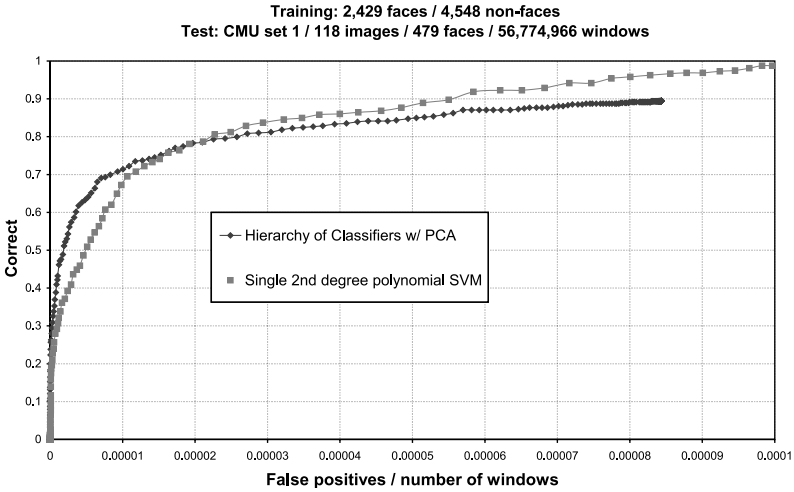

Fig. 9. ROC curves of the hierarchical system and the original system with a single second-degree polynomial SVM for the CMU set 1.

Table 2
Average computing time per 320×240 image for various face detection systems. Each image was processed at five different scales to detect faces at resolutions between 30×30 and 70×70 pixels

System	Time/image (ms)	Speed-up factor
Single SVM	86,875	—
Single SVM with feature reduction	23,383	3.7
Hierarchy of SVMs	391	222.2
Hierarchy of SVMs with feature reduction	259	335.4

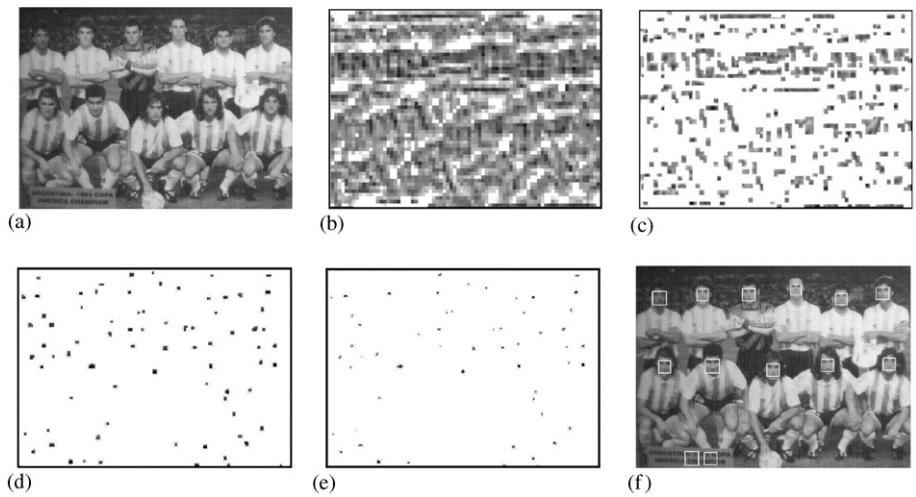

Fig. 10. Detections at each level of the hierarchy. (a) Original Image. (b–e) Outputs of layers two to five. Dark pixels represent high outputs of the classifier, white areas have been removed by previous layers. (f) Final detection result computed by layer six.

Fig. 11. Face detection with the hierarchical system.

of an image we arranged five SVM classifiers with increasing computational complexity in a hierarchical structure. We proposed an iterative algorithm for automatically building and training a hierarchical system of classifiers. To further speed-up the detection system we applied feature reduction to the non-linear SVM at the top level. This was accomplished by ranking and then selecting PCA gray features according to a classification criterion that was derived from learning theory. Applying these methods to face detection, we removed 99% of the features without loss in classification performance. Experiments show that the combination of feature selection and hierarchical classification results in a speed-up factor of 335 while maintaining classification accuracy. In the current system image preprocessing and feature extraction account for 90% of the runtime. In future work we will explore alternative feature extraction and preprocessing techniques to further speed-up the detection.

8. Summary

We present speed-up methods for a face detection system based on hierarchical classification and feature reduction. To quickly remove large background parts of an image we arrange five Support Vector Machine (SVM) classifiers with increasing computational complexity in a hierarchical structure. We propose an iterative algorithm for automatically building and training a hierarchical system of classifiers. To further speed-up the detection system we apply feature reduction to the non-linear SVM at the top level. This is accomplished by ranking and then selecting PCA gray features according to a classification criterion that was derived from learning theory. Applying these methods to face detection, we remove 99% of the features without loss in classification performance. Experiments show that the combination of feature selection and hierarchical classification results in a speed-up factor of 335 while maintaining classification accuracy.

References

[1] B. Heisele, T. Poggio, M. Pontil, Face detection in still gray images, A.I. Memo 1687, Center for Biological and Computational Learning, MIT, Cambridge, MA, 2000.
[2] A. Rosenfeld, G.J. Vanderbrug, Coarse-fine template matching, IEEE Trans. Syst. Man Cybernet. 2 (1977) 104–107.
[3] P.J. Burt, Smart sensing within a pyramid vision machine, Proc. IEEE 76 (8) (1988) 1006–1015.
[4] J. Edwards, H. Murase, Appearance matching of occluded objects using coarse-to-fine adaptive masks, Proc. IEEE Comput. Vision Pattern Recognition, Rierto Rico, 1997, pp. 533–539.
[5] H.A. Rowley, Neural network-based face detection, Ph.D. Thesis, CMU, School of Computer Science, Pittsburgh, 1999.
[6] A. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artif. Intell. 10 (1997) 245–271.
[7] R. Kohavi, Wrappers for feature subset selection, Artificial Intelligence (special issue on relevance) 97 (1995) 273–324.
[8] P. Viola, M. Jones, Robust real-time face detection, in: Proceedings of Eighth International Conference on Computer Vision, Vancouver, Vol. 20(11), 2001, pp. 1254–1259.
[9] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[10] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio, Pedestrian detection using wavelet templates, in: IEEE Conference on Computer Vision and Pattern Recognition, San Juan, 1997, pp. 193–199.
[11] K.-K. Sung, Learning and example selection for object and pattern recognition, Ph.D. Thesis, MIT, Artificial Intelligence Laboratory and Center for Biological and Computational Learning, Cambridge, MA, 1996.
[12] H.A. Rowley, S. Baluja, T. Kanade, Rotation invariant neural network-based face detection, Computer Science Technical Report CMU-CS-97-201, CMU, Pittsburgh, 1997.
[13] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, V. Vapnik, Feature selection for support vector machines, in: T.K. Leen, T.G. Diettrich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, Vol. 13, MIT Press, Cambridge, MA, 2001, pp. 668–674.

About the Author—BERND HEISELE received the M.Sc. and Ph.D. degrees in electrical engineering from the University of Stuttgart, Stuttgart, Germany, in 1993 and 1998, respectively. In 1999 he was awarded a postdoctoral fellowship by the DFG in Germany. From 1999 to 2001, he worked as a postdoctoral researcher at the Center for Biological and Computational Learning, Massachusetts Institute of Technology, Cambridge. He subsequently joined Honda and is currently heading the Honda Research Laboratory in Cambridge where he is conducting research in computer vision. His research interests are learning-based object detection/recognition and motion analysis in image sequences.

About the Author—THOMAS SERRE received the M.Sc. degree in image processing from the Ecole Nationale Supérieure des Télécommunications de Bretagne, Brest, France in 2000 and the M.Sc. degree in signal processing from the University of Rennes, Rennes, France the same year. From 2000 to 2001, he worked as a visiting scientist at the Center for Biological and Computational Learning (CBCL), Massachusetts Institute of Technology (MIT), Cambridge. He subsequently started a Ph.D. program at CBCL in the department of Brain and Cognitive Sciences at MIT. His research interests include biological and computational object recognition.

About the Author—SAM PRENTICE is pursuing the B.S. degree in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge. From 2000 to 2002 he worked as an undergraduate researcher at the Center for Biological and Computational Learning, MIT, and the Honda Research Laboratory, Boston, studying speed-up methods for object categorization with emphasis on face detection. He will complete his undergraduate studies in 2004.

About the Author—TOMASO POGGIO received the doctorate degree in theoretical physics from the University of Genoa in 1970. From 1971 to 1981, he held a tenured research position at the Max Planck Institute, after which he became a professor at Massachusetts Institute of Technology (MIT). Currently, he is Professor in the Department of Brain and Cognitive Sciences at MIT and a member of the Artificial Intelligence Laboratory. He is doing research in computational learning and vision at the MIT Center for Biological and Computational Learning, of which he is a co-director.