Feature Reduction and Hierarchy of Classifiers for Fast Object Detection in Video Images

Bernd Heisele[†‡] Thomas Serre[†] Sayan Mukherjee[†] Tomaso Poggio[†]
[†]Center for Biological and Computational Learning, M.I.T.
Cambridge, MA, USA
[‡]Honda R&D Americas, Inc., Boston, MA, USA
{heisele,serre,sayan,tp}@ai.mit.edu

Abstract

We present a two-step method to speed-up object detection systems in computer vision that use Support Vector Machines (SVMs) as classifiers. In a first step we perform feature reduction by choosing relevant image features according to a measure derived from statistical learning theory. In a second step we build a hierarchy of classifiers. On the bottom level, a simple and fast 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. Experiments with a face detection system show that combining feature reduction with hierarchical classification leads to a speed-up by a factor of 170 with similar classification performance.

1 Introduction

Most object detection tasks in computer vision are computationally expensive because of a) the large amount of input data that has to be processed and b) the use of complex classifiers that are robust against pose and illumination changes. Speeding-up the classification is therefore of major concern when developing systems for real-world applications. In the following we investigate two methods for speed-ups: feature reduction and hierarchical classification.

In [3] 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. However, searching an image for faces at different scales took several minutes on a PC—far too long for most real-world applications. One way to speed-up is to reduce the number of features.

There are basically two types of feature selection methods in the literature: filter and wrapper methods [1]. Filter methods are preprocessing steps performed independently of the classification algorithm or its error criteria; PCA is an example of a filter method. Wrapper methods attempt to search through the space of feature subsets using the criterion of the classification algorithm to select the optimal feature subset. Wrapper methods can provide more accurate solutions than filter methods [5], but in general are more computationally expensive. We present a new wrapper method to reduce the dimensions of both input and feature space of an SVM. An alternative approach for speeding-up SVM classification has been proposed in [7] by reducing the number of support vectors.

Feature reduction is a generic tool that can be applied to any classification problem. When dealing with a specific classification task we can use prior knowledge about the type of data to speed-up classification. Two assumptions hold for most vision-based object detection tasks: a) The vast majority of the analyzed patterns in an image belongs to the background class and b) most of the background patterns can be easily distinguished from the objects. Based on these two assumptions 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 falls into the framework of coarse-to-fine template matching [8, 2] and is also related to biologically motivated work on attention-based vision [4].

More recently a cascade of linear classifiers that have been trained using AdaBoost has been proposed in [12] for frontal face detection. This idea is related to ours in the sense that it combines hierarchical classification with feature selection. However, in our approach the complexity of the classifiers in the hierarchy is not only controlled by the number of features (image resolution) but also by the class of decision functions (i.e. class of SVM kernel functions). The bottom level of our hierarchy consists of a linear classifier that operates on low resolution patterns (9×9) while the top level consists of a non-linear classifier operating on higher resolution patterns (19×19).

In Section 2 we give a brief overview on SVM theory and describe the training and test data used in our experiments. In Section 3 we rank and select features in the input space. Feature selection in the feature space of the classifier is described in Section 4. In Section 5 we present the hierarchical structure of classifiers. The paper is concluded in Section 6.

2 Background

2.1 Support Vector Machine Theory

An SVM [11] 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:

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

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:

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

under constraints $\sum_{i=1}^{\ell} \alpha_i y_i = 0$ and $C \geq \alpha_i \geq 0, \;\; i = 1, ..., \ell$. An upper bound on the expected error probability EP_{err} of an SVM classifier is given by [11]:

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

where $M = \frac{1}{W(\alpha^0)}$ is the distance between the support vectors and the separating hyperplane and R is the radius of the smallest sphere including all points $\Phi(\mathbf{x}_1), ..., \Phi(\mathbf{x}_\ell)$ 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 [6, 3]. 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:

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

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 [11]. 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 $p = \frac{(n+3)n}{2}$ and is given by $\mathbf{x}^* = (\sqrt{2}x_1, \cdots, \sqrt{2}x_n, x_1^2, \cdots, x_n^2, \sqrt{2}x_1x_2, \cdots, \sqrt{2}x_{n-1}x_n)$. Thus the number of multiplications required for projecting the input vector into the feature space and for computing the decision function is:

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

From Eq. (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.

2.3 Training and test sets

In our experiments we used one training and two test sets. The positive training set contained 2,429 19×19 faces. The negative training set contained 4,548 randomly selected non-face patterns. The relatively small size of the training set affects the classification performance. Experiments in [3] show that for a given classifier the false positive rate can be reduced by a factor of 10 by increasing the training set using bootstrapping methods. In this paper the main goal was to speed-up a given classifier without loss of classification performance. We opted for a small training set in order to save time during training classifiers in our numerous experiments.

The test set was extracted from the CMU test set 1[1]. We extracted 472 faces and 23,570 non-face patterns. The non-face patterns were selected by a linear SVM classifier as the

[1]The test set is a subset of the CMU test set 1 [9] which consists of 130 images and 507 faces. We excluded 12 images containing line-drawn faces and non-frontal faces.

non-face patterns most similar to faces. The final evaluation of our system was performed on the entire CMU test set 1, containing 118 images. Processing all images at different scales resulted in about 57,000,000 analyzed 19×19 windows.

3 Dimension reduction in the Input Space

3.1 Ranking Features in the Input Space

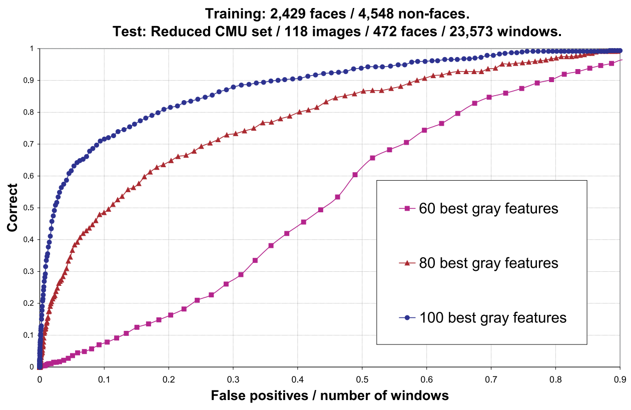
In [13] a gradient descent method is proposed to rank the input features by minimizing the bound of the expectation of the leave-one-out error of the classifier. The basic idea is to re-scale the n-dimensional input space by a n×n diagonal matrix σ such that $\frac{R^2}{M^2}$ is minimized. The new mapping 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:

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

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

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

subject to constraints $\sum_{i=1}^{\ell} \alpha_i y_i = 0$ and $C \geq \alpha_i \geq 0, \; i = 1, ..., \ell$. We solve for α and σ using an iterative procedure: We initialize σ as a vector of ones and then solve Eq. (7) and for the margin and radius. Using the values for α and β from the above equations 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_i 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 [10]. Additionally we performed tests with PCA gray features that we obtained by projecting the data points into the 283 dimensional eigenvector space. PCA was computed on a the combined set of positive and negative sets. We computed one iteration of the algorithm in all of our experiments. The tests were performed on the small test set for 60, 80 and 100 ranked features. The ROC curves for second-degree polynomial SVMs are shown in Fig. 1. For 100 features there is no difference between gray and PCA gray features. For 80 and 60 features, however, the PCA gray features gave clearly better results. For this reason we focused in the remainder of the paper on PCA features only. An interesting observation is that the ranking of the PCA features obtained by the above described gradient descent method was similar to the ranking by decreasing eigenvalues.



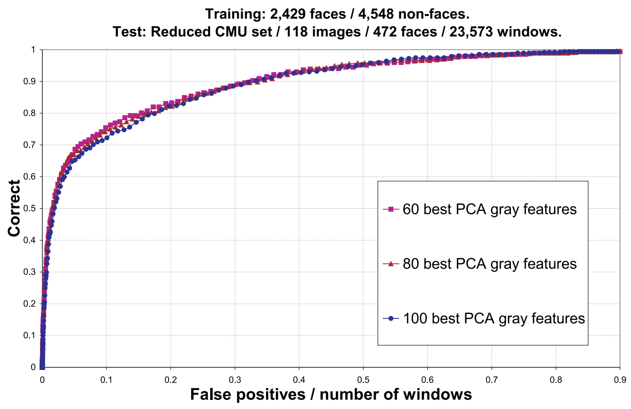


Figure 1. ROC curves for gray (top) and PCA gray (bottom) features with the 60, 80 and 100 best ranked features.

3.2 Selecting Features in the Input Space

In Section 3.1 we ranked the features according to their scaling factors σ_i. Now the problem is to determine a subset of the ranked features $(x_1, x_2, ..., x_n)$. This problem can be formulated as finding the optimal subset of ranked features $(x_1, x_2, ..., x_{n^*})$ among the n possible subsets, where 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 used again the bound on the expected error probability.

$$EP_{err} \leq \frac{1}{\ell} E\left(\frac{R^2}{M^2}\right) \quad (8)$$

To simplify the computation of our algorithm and to avoid solving a quadratic optimization problem in order to compute the radius R, we approximated[2] R^2 by $2p$ where p is

[2]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

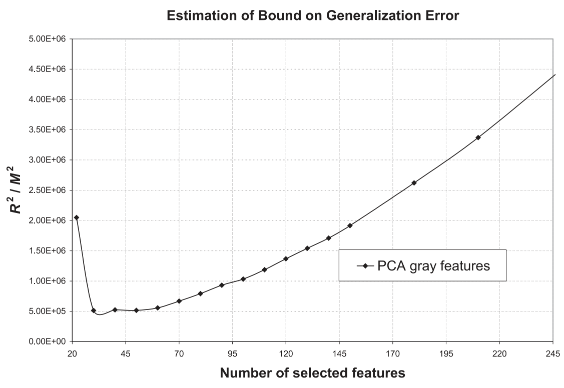


Figure 2. Approximation of estimated bound on the expected error versus number of principal components. The values on the y-axis are not normalized by the number of training samples.

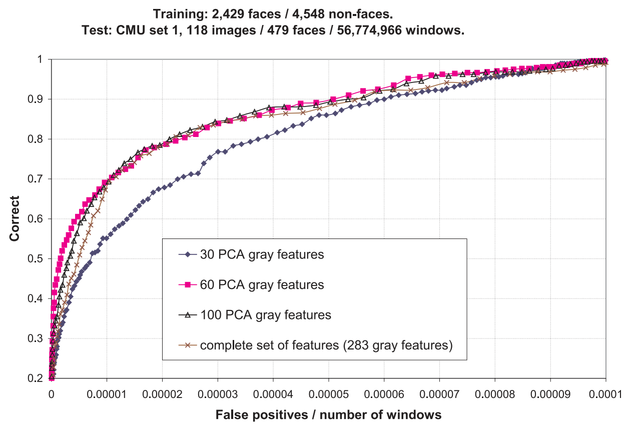


Figure 3. ROC curves for different number of PCA gray features.

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:

$$EP_{err} \leq \frac{1}{\ell}\, 2p\, E\left(W^2(\alpha^0)\right) \leq \frac{1}{\ell}\, n^*(n^*+3)\, E\left(W^2(\alpha^0)\right) \tag{9}$$

where n^* is the number of selected features[3]. The estimated bound on the expected error is shown in Fig. 2. We had no training error for more than 22 selected features. The estimated bound on the expected error shows a plateau between 30 to 60 features, then it increases steadily.

The bound in Eq. (8) is considered to be a loose bound on the expected error. To check if the bound is of practical use for selecting the number of features we performed tests on the CMU test set. In Fig. 3 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. 2. However, the error on the test set does not change significantly for more than 70 features although the estimated bound on the expected error shown in Fig. 2 increases. Probably because our bound gets looser with increasing number of features through the approximation of R by the dimensionality of the feature space.

4 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.

$$f(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}) + b = \sum_{i=1}^{\ell} \alpha_i^0 y_i K(\mathbf{x}_i, \mathbf{x}) + b \tag{10}$$

with $\mathbf{w} = (w_1, ..., w_p)$. 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 $p = \frac{(n+3)n}{2}$ is given by $\mathbf{x}^* = (\sqrt{2}x_1, \cdots, \sqrt{2}x_n, x_1^2, \cdots, x_n^2, \sqrt{2}x_1x_2, \cdots, \sqrt{2}x_{n-1}x_n)$. The contribution of a feature x_k^* to the decision function in Eq. (10) 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 k-th 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

$$E(S) = \frac{1}{s} \sum_i |f(\mathbf{x}_i) - f_S(\mathbf{x}_i)| \tag{11}$$

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. 4 show that ranking by the weighted components of $\mathbf{w}$ lead to a faster convergence of $E(S)$ from Eq. (11) towards 0. Fig. 5 shows the ROC curves for 500 and 1000

bound by $\sqrt{2p}$.

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

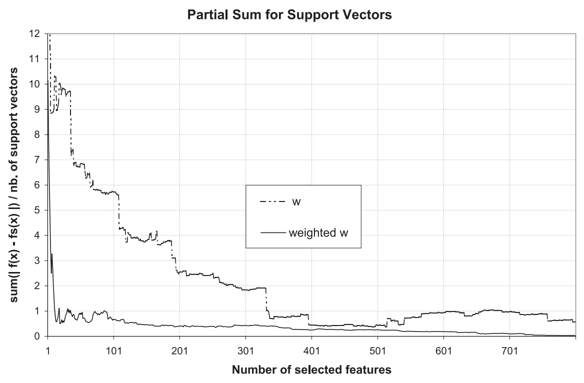


Figure 4. Classifying support vectors with a reduced number of features. The x-axis shows the number of features, the y-axis is the mean absolute difference between the output of the SVM using all features and the same SVM using the S first features only. The features were ranked according to the components and the weighted components of the normal vector of the separating hyperplane.

features. As a reference we added the ROC curve for a second-degree SVM trained on the original 283 gray features. This corresponds to $\frac{(283+3)283}{2} = 40,469$ 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.

5 Hierarchy of classifiers

5.1 System Overview

In most object detection problems the majority of analyzed image patches belongs to the background class. Only a small percentage of these patches look similar to objects and require a highly accurate classifier to avoid false classifications. For this reason we developed a 3-level hierarchy of classifiers where the computational complexity of the classifiers increases with each level. By propagating only those patterns that were classified as faces, we quickly decrease the amount of data when going up the hierarchy. The bottom level of our hierarchy consisted of a linear SVM that was trained on 9×9 face images. On the second level we increased the image resolution by a factor of two (19×19 face patterns) but kept the linear kernel. On the third level we finally used our best classifier, a nonlinear SVM with a second-degree polynomial kernel that was trained on 19×19 images. This classifier is highly sensitive to translation. If a face is not centered in the classification window it is likely to be classified as a non-face. In order not to miss any faces we search for faces in a small neighborhood around each detection location that was propagated from the second level. This means that we analyze 16 patterns on level three for each pattern that was classified as a face on level two. Fig. 9 gives an impression of the performance of the three individual classifiers; shown are the detection results for images from the CMU test set 1.

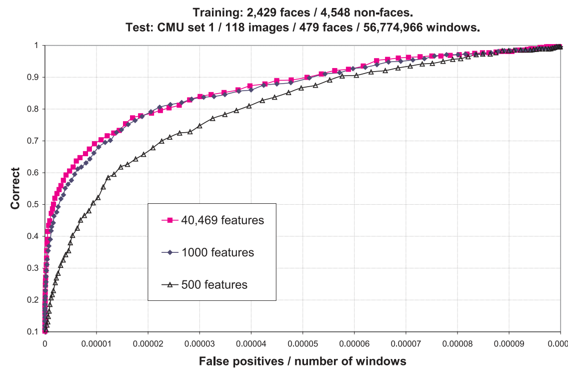


Figure 5. ROC curves for different dimension of the feature space.

5.2 Experiments

All three classifiers of our hierarchical system were trained on the same training set of 2,429 faces and 4,548 randomly selected non-face patterns. To train the low-resolution classifier in the first layer we downscaled the images from 19×19 to 9×9 pixels. The classifiers in the first two layers were trained on the gray value features described in Section 2.3. The third classifier was trained on PCA gray features which were determined by the feature selection techniques described in Sections 3 and 4 (1,000 features in the feature space determined from 60 PCA gray features of the input space). The ROC curves of the individual classifiers are shown in Fig. 6 for CMU test set 1. In Fig. 8 we show the data flow through the hierarchy for the CMU test set 1. The first classifier removes more than 90% of the background. The final classifier is most selective, it classifies more than 99% of its input patterns as non-faces. In Fig. 7 we compare the ROC curve of the 3-level system with the ROC curve of the original single SVM classifier with second-degree polynomial kernel. The performances are similar. The average computing time for a 320×240 image is shown in Table 1. We achieved a speed-up by a factor of 170 compared to the original system.

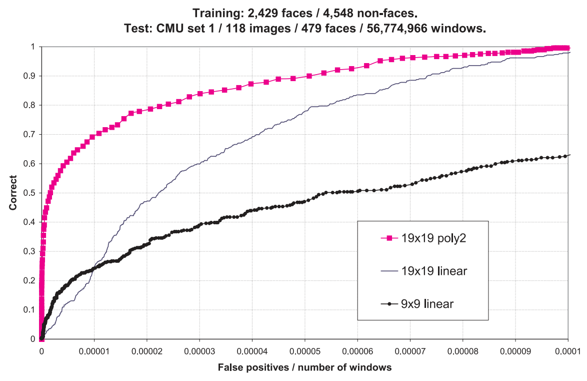


Figure 6. ROC curves for the three classifiers of the hierarchical system for the CMU test set 1.

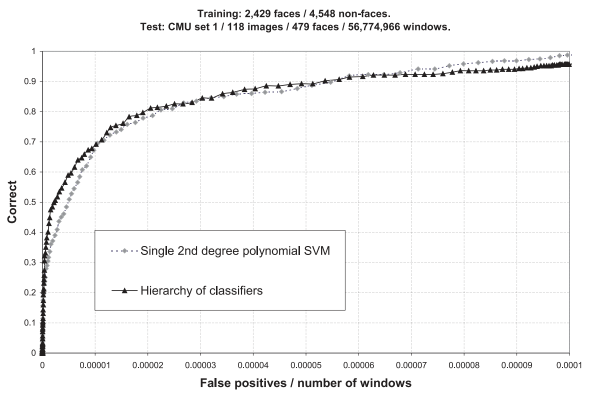


Figure 7. ROC curve of the 3 level hierarchical system and the original system for the CMU test set 1.

6 Conclusion and Future Work

In this paper we presented speed-up methods for object detection systems based on feature reduction and hierarchical classification. The feature reduction was done by ranking and then selecting PCA gray features according to a classification criterion that was derived from learning theory. Applied to a face detection system we could remove 99% of the original features without loss in classification performance. To quickly remove large background parts of an image we arranged three classifiers with increasing computational complexity in a hierarchical structure. Experiments with a face detection system show that the combination of feature selection and hierarchical classification speeds-up the system by a factor of 170 while maintaining the classification accuracy. In future work we will run experiments on larger training sets, apply feature reduction to all levels of the hierarchical classifier, and explore ways of finding the optimal number of levels. We will also perform experiments with hierarchical training where each classifier is trained on the outputs of the classifier of the previous level.

Acknowledgements

The authors would like to thank S. Prentice for helping with the experiments . The research was partially sponsored by DARPA under contract No. N00014-00-1-0907, and National Science Foundation under contract No. IIS-9800032. Additional support was provided by the DFG, Eastman Kodak, and Compaq.

References

[1] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 10:245–271, 1997.

[2] P. J. Burt. Smart sensing within a pyramid vision machine. *Proc. of the IEEE*, 76(8):1006–1015, 1988.

[3] B. Heisele, T. Poggio, and M. Pontil. Face detection in still gray images. A.I. memo 1687, Center for Biological and Computational Learning, MIT, Cambridge, MA, 2000.

[4] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Analysis and Machine Vision*, 20(11):1254–1259, 1998.

[5] R. Kohavi. Wrappers for feature subset selection. *Artificial Intelligence, special issue on relevance*, 97:273–324, 1995.

[6] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 193–199, San Juan, 1997.

[7] S. Romdhani, P. Torr, B. Schoelkopf, and A. Blake. Computationally efficient face detection. *Proc. ICCV*, II:695–700, 2001.

[8] A. Rosenfeld and G. J. Vanderbrug. Coarse-fine template matching. *IEEE Transactions on Systems, Man and Cybernetics*, 2:104–107, 1977.

[9] H. A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. Computer Scienct Technical Report CMU-CS-97-201, CMU, Pittsburgh, 1997.

[10] K.-K. Sung. *Learning and Example Selection for Object and Pattern Recognition*. PhD thesis, MIT, Artificial Intelligence Laboratory and Center for Biological and Computational Learning, Cambridge, MA, 1996.

[11] V. Vapnik. *Statistical learning theory*. John Wiley and Sons, New York, 1998.

[12] P. Viola and M. Jones. Robust real-time face detection. *Proc. ICCV*, 20(11):1254–1259, 2001.

[13] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for support vector machines. In *Advances in Neural Information Processing Systems 13*, 2001.

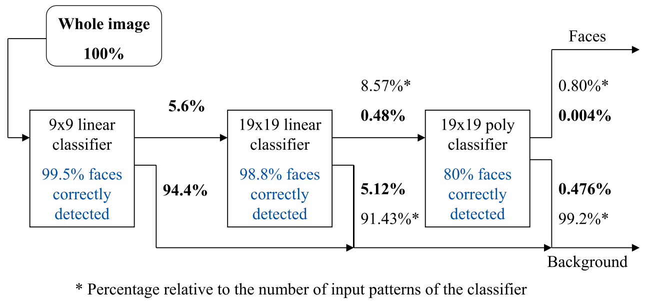

Whole image
100%
9x9 linear classifier
99.5% faces correctly detected
5.6%
94.4%
19x19 linear classifier
98.8% faces correctly detected
8.57%*
0.48%
5.12%
91.43%*
19x19 poly classifier
80% faces correctly detected
Faces
0.80%*
0.004%
0.476%
99.2%*
Background
* Percentage relative to the number of input patterns of the classifier

Figure 8. Data flow for the 3-level hierarchy of classifiers determined on the CMU test set 1.

System	Typical detection time	Speed-up factor
Single 2^{nd} degree polynomial SVM	271 s	–
Single 2^{nd} degree polynomial SVM + Feature reduction	63.8 s	4.25
3-Level hierarchy + Feature reduction	1.6 s	170

Table 1. Computing time for a 320×240 image processed on a dual Pentium III with 733 MHz. The original image was rescaled in 5 steps to detect faces at resolutions between 26×26 and 60×60 pixels.

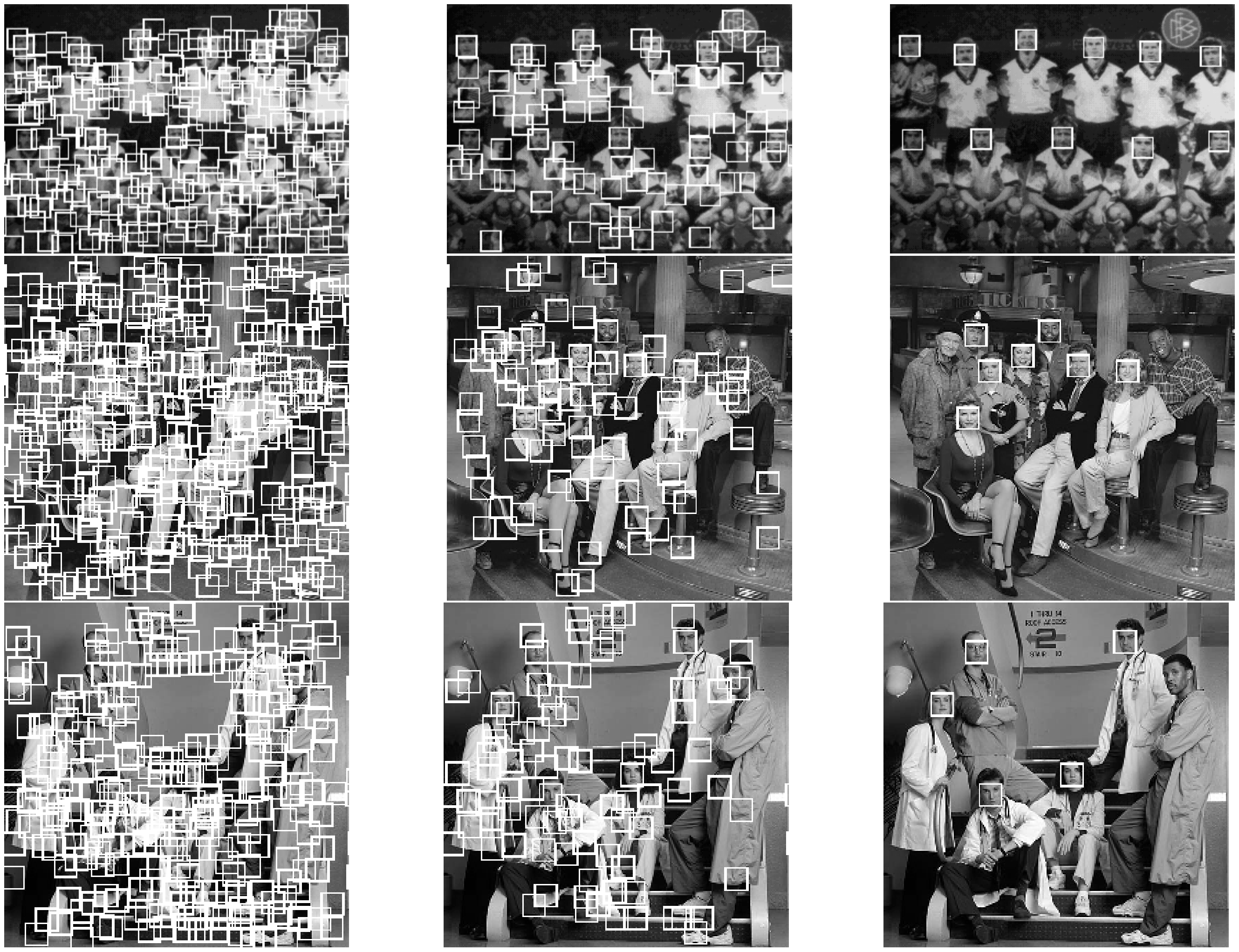


Figure 9. Detections at each level of the hierarchy.