Computationally Efficient Face Detection

Sami Romdhani, Philip Torr, Bernhard Schölkopf, Andrew Blake
Microsoft Research Ltd.
1 Guildhall Street
Cambridge, UK
romdhani@informatik.uni-freiburg.de {*philtorr,bsc,ablake*}*@microsoft.com*

Abstract

This paper describes an algorithm for finding faces within an image. The basis of the algorithm is to run an observation window at all possible positions, scales and orientation within the image. A non-linear support vector machine is used to determine whether or not a face is contained within the observation window. The non-linear support vector machine operates by comparing the input patch to a set of support vectors (which can be thought of as face and anti-face templates). Each support vector is scored by some non-linear function against the observation window and if the resulting sum is over some threshold a face is indicated. Because of the huge search space that is considered, it is imperative to investigate ways to speed up the support vector machine. Within this paper we suggest a method of speeding up the non-linear support vector machine. A set of reduced set vectors (RV's) are calculated from the support vectors. By considering the RV's sequentially, and if at any point a face is deemed too unlikely to cease the sequential evaluation, obviating the need to evaluate the remaining RV's. The idea being that we only need to apply a subset of the RV's to eliminate things that are obviously not a face (thus reducing the computation). The key then is to explore the RV's in the right order and a method for this is proposed.

1. Introduction

In this paper we consider the problem of face detection within a large collection of images, such as a large photographic database, images bandied about in emails or displayed on the internet. We consider the most general problem with no constraint on the position of the face, furthermore we allow the images to be monochrome or colour so that colour information alone cannot be used to reduce the search (leaving the exploration of colour cues to others).

This is a well researched problem and there have been a large number of different approaches to it. The most successful have included those of Osuna and Girosi [3] who applied support vectors (SV's) to the problem, that of Rowley *et al* [5] who used a neural network, and that of Schneiderman and Kanade [6] who pursued a maximum likelihood approach based on histograms of feature outputs. The one common thing to all these methods is that they are all based on running a 20×20 pixel observation window across the image at all possible locations, scales and orientations. This involves a high degree of computation as (a) the observation window is a 400 dimensional vector that has to be classified in a very non-linear space (b) there are hundreds of thousands of positions to search.

Within this paper we follow the support vector machine approach of Osuna and Girosi [3], our new contribution being the sequential application of the support vectors to speed up the algorithm, and an algorithm to determine the order of this evaluation. Nonlinear Support Vector Machines are known to lead to excellent classification accuracies in a wide range of tasks [7, 10], including face detection [3]. They utilize a set of support vectors to define a boundary between two classes, this boundary depending on a kernel function that defines a distance between two vectors. They are, however, usually slower classifiers than neural networks. The reason for this is that their run-time complexity is proportional to the number of SVs, i.e. to the number of training examples that the SVM algorithm utilizes in the expansion of the decision function. Whilst it is possible to construct classification problems, even in high-dimensional spaces, where the decision surface can be described by two SVs only, it is normally the case that the set of SVs forms a substantial subset of the whole training set. This is the case for face detection where several hundred support vectors can be needed.

There has been a fair amount of research on methods for reducing the run-time complexity of SVMs [2, 8]. In the present article, we employ one of these methods and adapt it to the case where the reduced expansion is not evaluated at once, but rather in a sequential way, such that in most cases a *very* small number of SVs are applied.

The paper is organised as follows: In Section 2 the general theory of support vector machines is reviewed with emphasis on non-linear support vector machines. In Section 3 it is explained how to compute a set of reduced support vec-

0-7695-1143-0/01 $10.00 (C) 2001 IEEE

tors and how to deduce a suitable order for their evaluation. The training is explained in Section 4 and the face finding algorithm in Section 5. Results are given in Section 6 and conclusion plus avenues for future work suggested in Section 7.

2. Non-linear Support Vector Machines

Support Vector classifiers implicitly map the data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell) \in \mathcal{X} \times \{\pm 1\}$ (in our case, $\mathcal{X}$ is the 20×20 observation window being a 400 dimensional integer valued vector) into a dot product space F via a (usually nonlinear) map $\Phi : \mathcal{X} \to F, \quad \mathbf{x} \mapsto \Phi(\mathbf{x})$. F is often referred to as the *feature space*. Although F can be high-dimensional, it is usually not necessary to explicitly work in that space [1]. There exists a class of kernels $k(\mathbf{x}, \mathbf{x}')$ which can be shown to compute the dot products in associated feature spaces, i.e. $k(\mathbf{x}, \mathbf{x}') = (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}'))$. The SV algorithm computes a hyperplane which separates the data in F by a large margin. Once this geometrical problem is cast in terms of dot products, the kernel trick is used and thus all computations in F are reduced to the evaluation of the kernel. It can be shown that the resulting training problem consists of computing (for some positive value of the parameter C determining the trade-off between margin maximization and training error minimization)

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

$$\text{subject to } \; 0 \le \alpha_i \le C, \;\; i = 1, \ldots, \ell, \;\; \sum_{i=1}^{\ell} \alpha_i y_i = 0, \tag{2}$$

and that the solution has an expansion

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{\ell} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + b \right). \tag{3}$$

Those training examples $\mathbf{x}_i$ with $\alpha_i > 0$ are called *Support Vectors*.

Kernels commonly used include polynomials $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}')^d$, which can be shown to map into a feature space spanned by all order d products of input features, and the Gaussian RBF kernel

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|^2}{2\,\sigma^2} \right). \tag{4}$$

Performance-wise, they have been found to do similarly well; in the present paper, we focus on the latter of the two. This means that support vectors act as templates for faces and anti-faces, thus relating non-linear SV's to vector quantization.

3. Reduced Set Vectors

Assume we are given a vector $\Psi \in F$, expanded in images of input patterns $\mathbf{x}_i \in \mathcal{X}$,

$$\Psi = \sum_{i=1}^{N_x} \alpha_i \Phi(\mathbf{x}_i), \tag{5}$$

with $\alpha_i \in \mathbb{R}, \mathbf{x}_i \in \mathcal{X}$. To reduce the complexity of evaluating it, one can approximate it by a *reduced set* expansion [2]

$$\Psi' = \sum_{i=1}^{N_z} \beta_i \Phi(\mathbf{z}_i), \tag{6}$$

with $N_z \ll N_x$, $\beta_i \in \mathbb{R}$, and *reduced set vectors* $\mathbf{z}_i \in \mathcal{X}$. To this end, one can minimize [2]

$$\|\Psi - \Psi'\|^2 = \sum_{i,j=1}^{N_x} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i,j=1}^{N_z} \beta_i \beta_j k(\mathbf{z}_i, \mathbf{z}_j) - 2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_z} \alpha_i \beta_j k(\mathbf{x}_i, \mathbf{z}_j). \tag{7}$$

The key point of that method is that although Φ is not given explicitly, (7) can be computed (and minimized) in terms of the kernel.

The sequential approach used here requires an extension of the reduced set method, to compute a whole sequence of reduced set approximations

$$\Psi'_m = \sum_{i=1}^{m} \beta_{m,i} \Phi(\mathbf{z}_i), \tag{8}$$

for $m = 1, \ldots, N_z$. The reduced set vectors $\mathbf{z}_i$ and the coefficients $\beta_{m,i}$ are computed by iterative optimization [8]. For the first vector, we need to approximate $\Psi = \sum_{i=1}^{N_x} \alpha_i \Phi(\mathbf{x}_i)$ by $\Psi' = \beta \Phi(\mathbf{z})$. Minimizing the distance $\|\Psi - \Psi'\|^2$ between Ψ and Ψ', with respect to $\mathbf{z}$, β, to give the first reduced set vector $\mathbf{z}_1$ and its coefficient $\beta_{1,1}$, using the method set out in the appendix.

Recall that the aim of the reduced set algorithm is to approximate a vector Ψ as in equation (5) by an expansion of the type (6) with $N_z > 1$. The required higher order reduced set vectors $\mathbf{z}_i$, $i > 1$ and their coefficients β_i, are obtained in recursive fashion by defining a residual vector

$$\Psi_m = \Psi - \sum_{i=1}^{m-1} \beta_{m-1,i} \Phi(\mathbf{z}_i), \tag{9}$$

where Ψ is the original feature-space vector defined in (5). Then the procedure for obtaining the first reduced set vector $\mathbf{z}_1$ is repeated, now with Ψ_m in place of Ψ to obtain $\mathbf{z}_m$. However, the optimal β from this step is not used, instead

0-7695-1143-0/01 $10.00 (C) 2001 IEEE

optimal $\beta_{m,i},\ i = 1, \ldots, m$ are jointly computed [8]. Figure 1 demonstrates the effects on the classification boundary of sequential reduced set vector evaluation. Note that there is a law of diminishing returns, the first few RV's yielding the greatest increase in discrimination.

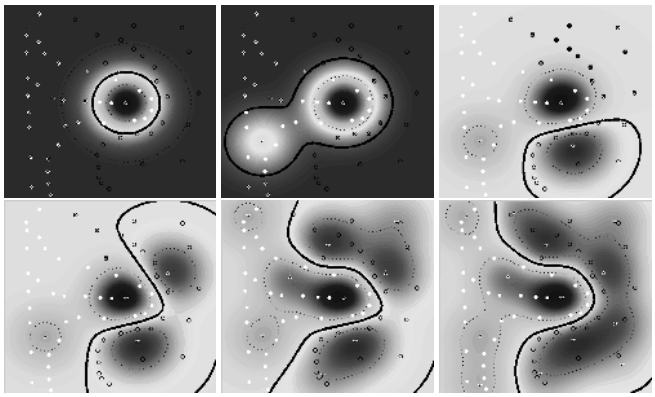

Figure 1: *The result of the sequential application of RV's (stars) to a classification problem, showing the result of using 1,2,3,4,9 and 13 RV's Darker regions indicate strong support for the classification.*

Thresholds. For any N_z, the obtained expansion can be plugged into the SVM decision function (3) to yield $f(\mathbf{x}) = \text{sgn}\left(\sum_{j=1}^{N_z} \beta_j k(\mathbf{x}, \mathbf{z}_j) + b\right)$. It is, however, not optimal to simply re-use the offset b stemming from the original SV machine. Reduced set approximations of decision functions can be improved by recomputing the thresholds b_j based on the training set or some validation set [8], to get

$$f_{N_z}(\mathbf{x}) = \text{sgn}\left(\sum_{j=1}^{N_z} \beta_j k(\mathbf{x}, \mathbf{z}_j) + b_{N_z}\right). \qquad (10)$$

This is especially true in the present setting, as will become clear in the following.

4. Training

Initially the SVM was trained on 3600 frontal face and 25000 non-face examples using Platt's Sequential Minimal Optimisation [4]. The kernel used was Gaussian (Equation 4) with a standard deviation σ of 3.5. The trade-off between margin maximization and training error minimization, was set to 1. The non-face patches were taken randomly on a set of 1000 images containing no faces. The SVM selected 1742 support vectors.

To improve the performance of the classifier a second bout of training was initiated: To decrease the number of false positives the face detector was applied on a new set of 100 images which did not contain any faces. This generated 110000 false positive patches which were then added to the training. To decrease the number of false negatives, virtual faces were generated and added to the training set. These virtual faces were computed by modifying the contrast or by adding an illumination plane to the faces of the original training set. This alleviates the need of computing a pre-processing at detection time and increase the run-time performance of our algorithm. The SVM was then retrained using this new training set which yielded 8291 support vectors. These were subsequently decreased to 100 reduced set vectors. Note that a retraining using the misclassifications of a previous training has been shown in [5] to produce a greatly improved classifier.

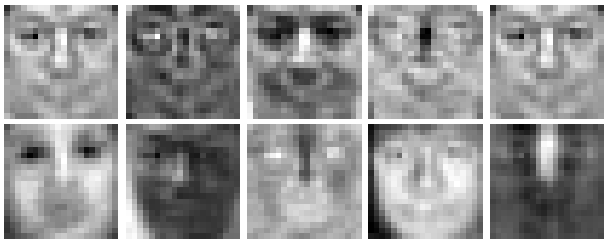

Figure 2: *First 10 reduced set vectors. Note that all vectors can be interpreted as either faces (e.g. the first one) or anti-faces (e.g. the second one)*

.

5. Face Detection by Sequential Evaluation

At detection time, *each* pixel of an input image is a potential face (a large number). To detect faces at different scales an image pyramid is constructed. If *w* and *h* are the width and the height of the input image and L and s the number of sub-sampling levels and the sub-sampling rate, respectively, the total number of patches to be evaluated is $N_p = \sum_{l=1}^{L} whs^{2(l-1)}$. Evaluating the full SVM or even the whole set of reduced vectors on all patches would be slow. A large portion of the patches can be easily classified using only a few reduced set vectors. Hence we propose the following *Sequential Evaluation* algorithm, to be applied to each overlapping patch $\mathbf{x}$ of an input image.

1. Set the hierarchy level to $m = 1$.
2. Evaluate $y_m = \text{sgn}\left(\sum_{j=1}^{m} \beta_{m,j} K_j + b_m\right)$ where $K_j = k(\mathbf{x}, \mathbf{z}_j)$.
3. - if $y_m < 0$, $\mathbf{x}$ is classified as a non-face and the algorithm stops.
   - if $y_m \geq 0$, m is incremented. If $m = N_z$ the algorithm stops, otherwise evaluation continues at step 2.

0-7695-1143-0/01 $10.00 (C) 2001 IEEE

4. if $y_j \geq 0$ and $j = N_z$, the full SVM is applied on the patch x, using equation 3. If the evaluation is positive the patch is classified as a face.

The main feature of this approach is that on average, relatively few kernels K_j have to be evaluated at any given image location — i.e., for most patches, the algorithm above stops at a level $j \ll N_z$. This speeds up the algorithm relative to the full reduced set (by more than an order of magnitude in the face classification experiments reported below). Note that in the case of gaussian kernels, the application of one reduced set vector amounts to a simple template matching operation.

Setting offsets. The offsets b_m are fixed to obtain a desired point on the R.O.C. for the overall sequential scheme. Suppose an overall false negative rate ν is required, then, given a "decay rate" α, we express ν as a geometric series by setting false negative rates ν_m for the mth level in the hierarchy to $\nu_j = \alpha\nu_{j-1}$ where $\nu_1 = \nu(1-\alpha)$. Now each b_m is fixed to achieve the desired ν_m over a validation set. The free parameter α can now be set to maximize the overall true positive rate over the validation set.

6. Results

Within this section the new sequential evaluation algorithm is tested for speed and accuracy.

Speed Improvement. At detection time, due to the sequential evaluation of the patches, very few reduced set vectors are applied. Figure 3 shows the number of reduced set vectors evaluated per patches for different methods (SVM, RSM and SRSM (Sequential Reduced Set Machine)), when the algorithm is applied to the photo in Fig 4. The Full SVM and the RSM evaluate all their support or reduced set vectors on all the patches, while the SRSM uses on average only 2.8 reduced set vectors per patch. Figure 4 shows the patches of an input image which remain after 1, 10, 20 and 30 sequential reduced set evaluations on an image with one face, figure 5 shows the results on an image with multiple faces.

Figure 7 shows the number of reduced set vectors used to classify each patch of an image. The intensities values of the pixels of the right image are proportional to the number of reduced set vectors used to classify the corresponding spot in the left image (note that the intensities are displayed at the center of the corresponding patches only). The uniform parts of the input image are easily rejected using a single reduced set vector, whereas the cluttered background requires more reduced set vectors. Note that very few patches needed all the reduced set vectors (only the patches containing the faces used all the reduced set vectors).

Accuracy. Figure 6 shows a comparison of the accuracy of the different methods. These R.O.C. were computed on

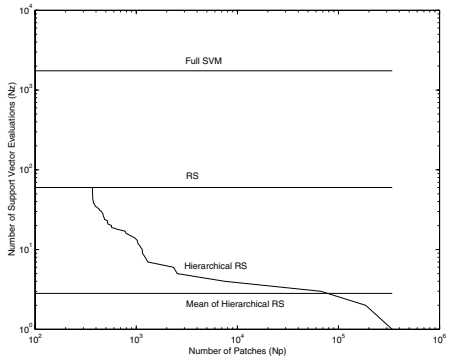

Figure 3: *Number of reduced set vectors used per patch for the full SVM (8291 support vectors), Reduced Set SVM and Sequential Reduced Set SVM (both at 100 reduced set vector)*

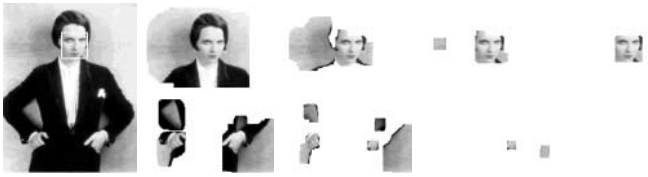

Figure 4: **From left to right:** *input image, followed by portions of the image which contain un-reject patches after the sequential evaluation of 1* (13.3% patches remaining)*, 10* (2.6%)*, 20* (0.01%) *and 30* (0.002%) *support vectors. Note that in these images, a pixel is displayed if it is part of any remaining un-rejected patch at any scale, orientation or position This explains the apparent discrepancy between the above percentages and the visual impression.*

a test set containing 800 faces and 5000 non-faces. The accuracy of the SRSM (100 reduced set vectors) is very similar to the accuracy of the full SVM (8291 support vectors) and the RS (100 reduced set vectors) which perform equally well.

To compare our system with others, we used the Rowley *et al.* [5] test set (which also includes the Sung *et al.* [9] and the Osuna *et al.* [3] test images). This set consists of 130 images containing 507 faces. We used a sub-sampling ratio of $s = 0.7$ and the input images were sub-sampled as long as their width and height was larger than 20 (i.e. the number of levels in the sub-sampling pyramid is $\min\left(\text{floor}\left(\frac{\log(20/w)}{\log 0.7}\right), \text{floor}\left(\frac{\log(20/h)}{\log 0.7}\right)\right)$ where w and h are, respectively, the width and the height of the input image). We obtained a detection rate of 80.7% with a false detection rate of 0.001%. These numbers are slightly worse than Rowley's, Sung's and Osuna's results, although they are hard to compare due to the fact that they pre-process the patches before feeding them into their classifier (histogram equalisation, background pixel removal and illumination gradient compensation). Our main objective was speed, hence no pre-processing was made. Secondly, we

0-7695-1143-0/01 $10.00 (C) 2001 IEEE

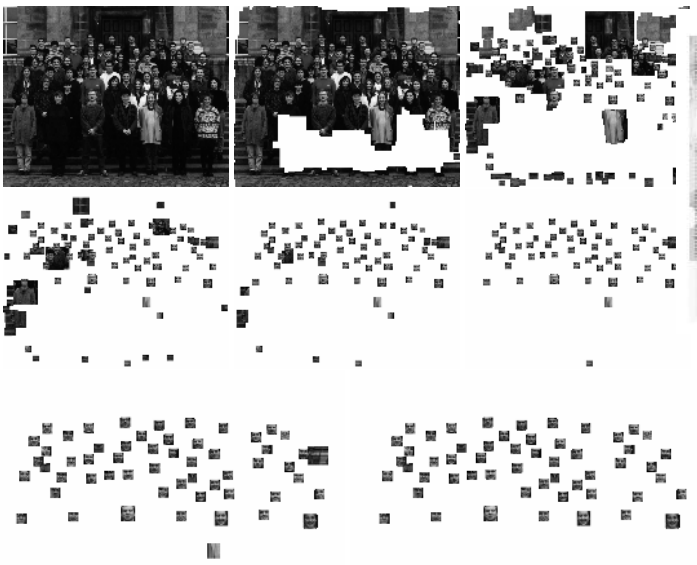

Figure 5: *Input image, followed by patches which remain after the evaluation of 1* (19.8% patches remaining)*, 10* (0.74%)*, 20* (0.06%) *and 30* (0.01%) *. . . 70* (0.007%)*support vectors. Note the comment in the caption of Fig 4.*

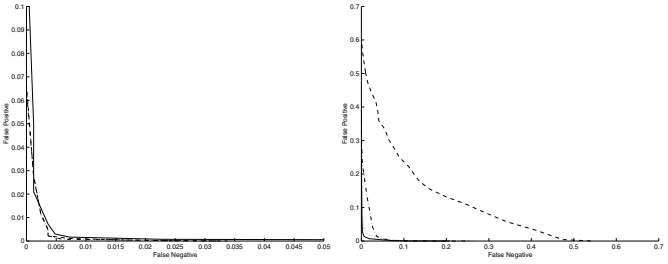

Figure 6: **Left:** *R.O.C. for the SVM using 8291 support vectors (dotted line), the RS using 100 reduced set vectors (dashed line) and SRSM using also 100 reduced set vectors (solid line). Note that the SVM and RS curves are so close that they are not distinguishable.* **Right:** *R.O.C. for an SRSM using 1 (dashed line), 2 (dash-dot line), 3 (dotted line) and 4 (solid line) reduced set vectors.*

used a different training set as their training set was partly proprietary. Speed figures are also hard to compare, but from the information given, we conjecture that the Osuna *et al.* RS system is comparable in speed to our RS system, which in turn is 30 times slower than our sequential evaluation system (32μs for the sequential evaluation, 1.2ms for the reduced set evaluation and 26ms for the full SVM per patch on a 500MHz Pentium).

7. Conclusion and Future Work

Pattern detection systems usually have to scan large images. Therefore, the greatest challenge in engineering systems for real-world applications is that of reducing computational

Figure 7: **Top left***: The intensity values of the pixels of the left image are proportional to the number of reduced set vectors used to classify their associated patches of the middle image. Light grey corresponds to the use of a single reduced set vector, black to the use of all the vectors.* **Top middle:** *153×263 middle image contains 76108 patches and was detected in* 2.58*s.* **Top right:** *A 601×444 image containing 518801 patches detected in 27.9s.* **Bottom Left:** *1280×1024 contains 2562592 patches and was detected in* 80.1*s.* **Bottom right:** *A 320×240 image containing 147289 patches detected in 10.4s (Note the false positives).*

complexity. Within this paper we have demonstrated computational savings in classification by the use of a sequential reduced support vector evaluation. There are several avenues for future research. (a) We have explored the use of the Gaussian kernel as a distance metric, however it may be possible to tailor the kernel to something much more suited to facial detection. (b) It may be that the criteria for choosing the reduced set of support vectors can be improved. At present the reduced set of support vectors is chosen to minimize (7), which affects classification error only indirectly. However, it might be advantageous to choose a reduced set that minimizes classification error directly. (c) It would be interesting to adapt the thresholds based on contextual information: for instance, if a face is detected in the image, this places strong priors on the scale and orientation of any other faces we expect to see. This could further speed up the detection. Finally, although the method has been implemented for the task of face detection, it could be readily applied to a wide class of other detection and classifications problems.

Acknowledgments

Thanks to Henry Rowley for assisting us and for providing test images. Thanks to Mike Tipping, Kentaro Toyama and Ben Bradshaw for useful conversations.

0-7695-1143-0/01 $10.00 (C) 2001 IEEE

References

[1] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proc. of the 5th ACM Workshop on Computational Learning Theory*, pages 144–152, Pittsburgh, PA, 1992. ACM Press.

[2] C. J. C. Burges. Simplified support vector decision rules. In *13th Intl. Conf. on Machine Learning*, pages 71–77, 1996.

[3] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *CVPR*, pages 130–136, 1997.

[4] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999. MIT Press.

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

[6] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to face and cars. In *CVPR*, pages 746–751, 2000.

[7] B. Schölkopf, C. J. C. Burges, and A. J. Smola. *Advances in Kernel Methods — Support Vector Learning*. MIT Press, Cambridge, MA, 1999.

[8] B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000 – 1017, 1999.

[9] K-K. Sung and T. Poggio. Example-based learning for view-based human face detection. In *Proceedings from Image Understanding Workshop*, Monterey, CA, November 1994.

[10] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, N.Y., 1995.

Computing the first reduced set vector. Now the algorithm for minimizing the distance $\|\Psi - \Psi'\|^2$ in section is presented. First, we minimize the distance between Ψ and the orthogonal projection of Ψ onto span $(\Phi(\mathbf{z}))$,

$$\left\| \frac{(\Psi \cdot \Phi(\mathbf{z}))}{(\Phi(\mathbf{z}) \cdot \Phi(\mathbf{z}))} \Phi(\mathbf{z}) - \Psi \right\|^2 = \|\Psi\|^2 - \frac{(\Psi \cdot \Phi(\mathbf{z}))^2}{(\Phi(\mathbf{z}) \cdot \Phi(\mathbf{z}))}. \tag{11}$$

i.e. maximize

$$\frac{(\Psi \cdot \Phi(\mathbf{z}))^2}{(\Phi(\mathbf{z}) \cdot \Phi(\mathbf{z}))}, \tag{12}$$

which can be expressed in terms of the kernel. Once the maximum is found, it is extended to the minimum of $\|\Psi - \Psi'\|^2$ by setting (cf. (11)) $\beta = (\Psi \cdot \Phi(\mathbf{z}))/(\Phi(\mathbf{z}) \cdot \Phi(\mathbf{z}))$.

For kernels which satisfy $k(\mathbf{z}, \mathbf{z}) = 1$ for all $\mathbf{z} \in \mathcal{X}$ (e.g. Gaussian kernels), (12) reduces to

$$(\Psi \cdot \Phi(\mathbf{z}))^2. \tag{13}$$

For the extremum, we have $0 = \nabla_{\mathbf{z}} (\Psi \cdot \Phi(\mathbf{z}))^2 = 2(\Psi \cdot \Phi(\mathbf{z})) \nabla_{\mathbf{z}} (\Psi \cdot \Phi(\mathbf{z}))$. To evaluate the gradient in terms of k, we substitute (5) to get the sufficient condition $0 = \sum_{i=1}^{N_x} \alpha_i \nabla_{\mathbf{z}} k(\mathbf{x}_i, \mathbf{z})$. For $k(\mathbf{x}_i, \mathbf{z}) = k(\|\mathbf{x}_i - \mathbf{z}\|^2)$ (e.g. Gaussians, or $(\|\mathbf{x}_i - \mathbf{z}\|^2 + 1)^c$ for $c = -1, -1/2$), we obtain $0 = \sum_{i=1}^{N_x} \alpha_i k'(\|\mathbf{x}_i - \mathbf{z}\|^2)(\mathbf{x}_i - \mathbf{z})$, leading to $\mathbf{z} = \frac{\sum_{i=1}^{N_x} \alpha_i k'(\|\mathbf{x}_i - \mathbf{z}\|^2) \mathbf{x}_i}{\sum_{i=1}^{N_x} \alpha_i k'(\|\mathbf{x}_i - \mathbf{z}\|^2)}$. For the Gaussian kernel $k(\mathbf{x}_i, \mathbf{z}) = \exp(-\|\mathbf{x}_i - \mathbf{z}\|^2/(2\sigma^2))$ we thus arrive at $\mathbf{z} = \frac{\sum_{i=1}^{N_x} \alpha_i \exp(-\|\mathbf{x}_i - \mathbf{z}\|^2/(2\sigma^2)) \mathbf{x}_i}{\sum_{i=1}^{N_x} \alpha_i \exp(-\|\mathbf{x}_i - \mathbf{z}\|^2/(2\sigma^2))}$, and devise an iteration

$$\mathbf{z}_{n+1} = \frac{\sum_{i=1}^{N_x} \alpha_i \exp(-\|\mathbf{x}_i - \mathbf{z}_n\|^2/(2\sigma^2)) \mathbf{x}_i}{\sum_{i=1}^{N_x} \alpha_i \exp(-\|\mathbf{x}_i - \mathbf{z}_n\|^2/(2\sigma^2))}. \tag{14}$$

The denominator equals $(\Psi \cdot \Phi(\mathbf{z}_n))$ and thus is nonzero in a neighbourhood of the extremum of (13), unless the extremum itself is zero. The latter only occurs if the projection of Ψ on the linear span of $\Phi(\mathcal{X})$ is zero, in which case it is pointless to try to approximate Ψ. Numerical instabilities related to $(\Psi \cdot \Phi(\mathbf{z}))$ being small can thus be approached by restarting the iteration with different starting values.

Without further detail, we note that (14) can be interpreted as a type of clustering which takes into account both positive and negative data [8].

Computing higher order reduced set vectors. Higher order, reduced set vectors $\mathbf{z}_m$, $m > 1$ are required and each z_m is computed from Ψ_m (defined above) as follows. Equation (14) is applied to Ψ_m (in place of Ψ) by expressing Ψ_m in its representation in terms of mapped input images: $\Psi_m = \sum_{i=1}^{\ell} \alpha_i \Phi(\mathbf{x}_i) - \sum_{i=1}^{m-1} \beta_i \Phi(\mathbf{z}_i)$, i.e. we need to set $N_x = \ell + m - 1$, $(\alpha_1, \ldots, \alpha_{N_x}) = (\alpha_1, \ldots, \alpha_\ell, -\beta_1, \ldots, -\beta_{m-1})$, and $(\mathbf{x}_1, \ldots, \mathbf{x}_{N_x}) = (\mathbf{x}_1, \ldots, \mathbf{x}_\ell, \mathbf{z}_1, \ldots, \mathbf{z}_{m-1})$.

At each step, we compute the optimal coefficients $\beta = (\beta_1, \ldots, \beta_m)$ using Proposition 1 (note that if the discrepancy Ψ_{m+1} has not yet reached zero, then K^z will be invertible).

Proposition 1 ([8]) *The optimal coefficients $\beta = (\beta_1, \ldots, \beta_m)$ for approximating $\Psi = \sum_{i=1}^{\ell} \alpha_i \Phi(\mathbf{x}_i)$ by $\sum_{i=1}^{m} \beta_i \Phi(\mathbf{z}_i)$ (for linearly independent $\Phi(\mathbf{z}_1), \ldots, \Phi(\mathbf{z}_m)$) in the 2-norm are given by*

$$\beta = (K^z)^{-1} K^{zx} \alpha. \tag{15}$$

Here, $K^z_{ij} := (\Phi(\mathbf{z}_i) \cdot \Phi(\mathbf{z}_j))$ and $K^{zx}_{ij} := (\Phi(\mathbf{z}_i) \cdot \Phi(\mathbf{x}_j))$.

The solution vector takes the form (6).

0-7695-1143-0/01 $10.00 (C) 2001 IEEE