

Image representations and feature selection for multimedia database search*

August 28, 2004

Theodoros Evgeniou¹

Technology Management Department, INSEAD,
Bd. de Constance, Fontainebleau 77300, France.

E-mail: *theodoros.evgeniou@insead.edu*

Massimiliano Pontil

Department of Information Engineering, University of Siena,
Via Roma 56, Siena, 53100, Italy.

E-mail: *pontil@dii.mit.edu*

Constantine Papageorgiou, Tomaso Poggio

Center for Biological and Computational Learning, MIT,
45 Carleton St., Cambridge, MA, 02142 USA.

E-mail: *{cpapa,tp}@ai.mit.edu*

*A shorter version of this paper was presented at ACCV' 2000.

¹Contact Author.

Abstract

The success of a multimedia information system depends heavily on the way the data is represented. Although there are “natural” ways to represent numerical data, it is not clear what is a good way to represent multimedia data, such as images, video, or sound. In this paper we investigate various image representations where the quality of the representation is judged based on how well a system for searching through an image database can perform - although the same techniques and representations can be used for other types of object detection tasks or multimedia data analysis problems. The system is based on a machine learning method to develop object detection models from example images that can subsequently be used for example to detect - search - images of a particular object in an image database. As a base classifier for the detection task we use support vector machines (SVM), a kernel based learning method. Within the framework of kernel classifiers we investigate new image representations/kernels derived from probabilistic models of the class of images considered, and present a new feature selection method which can be used to reduce the dimensionality of the image representation without significant losses in terms of the performance of the detection - search - system.

Index terms: Machine learning, object detection, support vector machines, image representation, multimedia data search.

1 Introduction

An important research issue in the field of multimedia data analysis is that of choosing the right representations for the data (images, sounds, video, etc). Whether it is for searching, indexing, comparison, etc, it is clear that the way the multimedia data are represented can significantly influence the performance for the various data analysis tasks. In this paper we focus on a particular task, namely that of detecting real-world objects in images for example for the purpose of searching in image databases, and we study the problem of choosing image representations for this task. Although we focus on object detection in images, a difficult multimedia analysis problem, the methods and experiments we present can be applied to any multimedia analysis task. We compare various standard and new representations in terms of how they influence the detection rate for new images and therefore the success of a multimedia search system. We also propose a new feature selection method that we use to create compact multimedia representations without significantly influencing the performance of the detection system.

Detection of real-world objects in images, such as faces, is a problem of fundamental importance in many areas of multimedia information systems (see [36] and references therein). An object detection system can be used as a component of an intelligent multimedia system: in this case, for example one seeks to find images containing specific objects, such as cars, faces, people, trees, etc, in an image or video database. Effective indexing into image and video databases relies on the detection of different classes of objects. Other image processing tasks also rely heavily on detection systems: for face - or other object recognition - the face must first be detected before being recognized; for autonomous navigation, obstacles and

landmarks need to be detected.

Detection of objects in images and video presents interesting challenges due to the significant variability in shape, color, and pose, and the typically high dimensionality of feature vectors. Another main problem is context, that is background. Since we are interested not simply in classifying the type of scene or image but also in finding images containing specific objects, the backgrounds against which the objects lie are unconstrained.

Initial work on object detection used template matching approaches with a set of rigid templates or hand-crafted parameterized curves [2, 39]. These approaches are difficult to extend to more complex objects such as people, since they involve a significant amount of prior information and domain knowledge. Other systems detect objects in video sequences focusing on using motion and 3D models or constraints to find people: Tsukiyama & Shirai [31], Leung & Yang[16], Hogg[11], Rohr [26], Wren, et al. [40], Heisele, et al. [10], McKenna & Gong [17]. In this paper we consider the issue of detection objects in single static images in unconstrained environments with cluttered backgrounds, while making no assumption on the scene structure.

In recent research the detection problem has been solved using learning-based techniques that are data driven. This approach was used by Sung and Poggio [30] and Vaillant, et al. [33] for the detection of frontal faces in cluttered scenes, with similar architectures later used by Moghaddam and Pentland [18], Rowley, et al. [27], and Osuna et al. [21]. In this paper we take the learning based approach to object detection. Because of the high dimensionality of feature vectors, we focus on the use of support vector machines (SVM) classifiers [34] as the core engine of these systems [23].

As already mentioned, a major issue in such systems is choosing an appropriate image

representation. We investigate the role of image representation for object detection using kernel machine classifiers such as SVM. In particular, we present experimental results comparing different image representations for both face and people detection. The image representations we used were either projections onto principal components (i.e. eigenfaces [18]), projections on wavelets, raw pixel values, or, finally, features derived from probabilistic models [30]. In this paper we compare these representations, and also link them through the choice of kernels in the SVM classifier.

The structure of the paper is as follows. Section 2 contains a description of the object detection system and a short review of SVM's. Section 3 presents experimental comparisons for face and people detection using different image representations. Subsequently, the effects of histogram equalization are discussed. Section 4 discusses the problem of feature selection, and presents a method for feature selection using SVM. Section 5 discusses the development of new image representations derived from probabilistic models. We conclude with some comments in Section 6.

2 Background

2.1 A trainable system for object detection

We outline the trainable system for object detection that we used in this work. The system is based on [23] and can be used to learn any class of objects. The overall framework has been motivated and successfully applied in the past [23]. The system consists of three parts:

1. A set of positive example images of the object class considered (e.g. images of frontal

faces) and a set of negative examples (e.g. any non-face image) are collected. The positive examples are scaled images of the object class that have been aligned so that they are all in approximately the same position in the image (see 1).

2. The images are transformed into (feature) vectors in a chosen representation (e.g. a vector of the size of the image with the values at each pixel location - below this is called the “pixel” representation).
3. The vectors (examples) are used to train a pattern classifier, in our case an SVM, to learn the classification task of separating positive from negative examples.

To detect objects in out-of-sample images, the system slides a fixed size window over an image and uses the trained classifier to decide which patterns show the objects of interest. At each window position, we extract the same set of features as in the training step and feed them into the classifier; the classifier output determines whether or not we highlight that pattern as an in-class object. To achieve multi-scale detection, we iteratively resize the image and process each image size using the same fixed size window.

Two closely related choices need to be made: the representation in the second stage, and the kernel of the SVM (see below) in the third stage. This paper focuses on these two choices.

In the experiments described below, we used the following data²:

- For the face detection systems, we used 2,429 positive images of frontal faces of size 19x19 (see figure 1), and 13,229 negative images randomly chosen from large images

²Available at <http://www.ai.mit.edu/projects/cbcl/>

not containing faces. The systems were tested on new data consisting of 105 positive images and around 4 million negative ones.

- For the people detection systems, we had 700 positive images of people of size 128x64 (see figure 1), and 6,000 negative images. The systems were tested on new data consisting of 224 positive images and 3,000 negative ones (for computational reasons, only for Figure 8 we used more test points: 123 positive and around 800,000 negative ones).



Figure 1: *Top row*: examples of images of faces in the training database. The images are 19x19 pixels in size. *Bottom row*: examples of images of people in the training database. The images are 128x64 pixels in size.

The performances are compared using Receiver Operating Characteristic (ROC) curves [23] generated by moving the hyperplane of the SVM solution by changing the threshold b (see below), and computing the percentage of false positives and false negatives for each choice of b . In the plots presented the vertical axis shows the percentage of positive test images correctly detected (1 - false negative percentage), while the horizontal axis shows one false positive detection per number of negative test images correctly classified.

2.2 Support vector machine classification

Support vector machines (SVM) are a technique to train classifiers ³ that is well-founded in statistical learning theory [34]. One of the main attractions of using SVM is that they are capable of learning in *sparse, high-dimensional spaces* with very few training examples. SVM accomplish this by minimizing a bound on the empirical error and the complexity of the classifier, at the same time.

Let $\boldsymbol{\alpha}$ be the vector of model parameters (in the case of SVM $\boldsymbol{\alpha}$ has the same dimensionality than the input examples - see below), $R_{emp}(\boldsymbol{\alpha})$ the empirical error, and $R(\boldsymbol{\alpha})$ expected error of the computed model. Statistical Learning Theory provides probabilistic bounds on the distance between the empirical and the expected risk for any model parameters. The bounds involves the number of examples ℓ and the *capacity* h of the parameter space, a quantity measuring the “complexity” of the space from where the model is computed. Appropriate capacity quantities are defined in the theory, the most popular one being the VC-dimension [35] or scale sensitive versions of it [15, 1]. The bounds have the following general form: with probability at least η

$$R(\boldsymbol{\alpha}) \leq R_{emp}(\boldsymbol{\alpha}) + \Phi\left(\frac{h}{\ell}, \eta\right),$$

where Φ is an increasing function of h/ℓ and η . For more information and the exact forms of the function Φ we refer the reader to [35, 34, 1]. Intuitively, if the capacity of the model parameters in which we minimize the empirical error is very large and the number of examples is small, then the distance between the empirical and expected risk can be large

³SVM can be used also for regression and density estimation problems [34].

and overfitting is very likely to occur.

This controlling of both the training error *and* the classifier's complexity has allowed SVM to be successfully applied to very high dimensional learning tasks such as face detection [21], 3-D object recognition [24], stop word detection in speech signals [19], and text categorization [14]. We will make use of this property of being able to apply SVM to very high dimensional classification problems.

The separating boundary implemented by an SVM is of the form:

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b \quad (1)$$

where ℓ is the number of training data $(\mathbf{x}_i, y_i) \in X \times \{-1, 1\}$, (for example in our case \mathbf{x}_i is an image and y_i is 1 if the image is in-class - i.e. an image of a face or a pedestrian - and -1 otherwise) α_i are nonnegative parameters learned from the data, and $K(\cdot, \cdot)$ is a kernel that defines a dot product between projections of the two arguments in some feature space [34, 38] where a separating hyperplane is then found (see below). For example, when $K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}$ is chosen (linear kernel), the separating surface is a hyperplane in the space X (input space). Classification of a new input \mathbf{x} is done by taking the sign of $f(\mathbf{x})$.

The main feature of SVM is that it finds, among all possible separating surfaces of the form (1), the one which maximizes the distance between the two classes of points (as measured in the feature space defined by K). Figure 2 shows this idea in the case of a linear kernel. The support vectors are the nearest points to the separating boundary and are the only ones (typically a small fraction of the training data) for which α_i in Equation (1) is positive. The distance of the closest one from the boundary is:

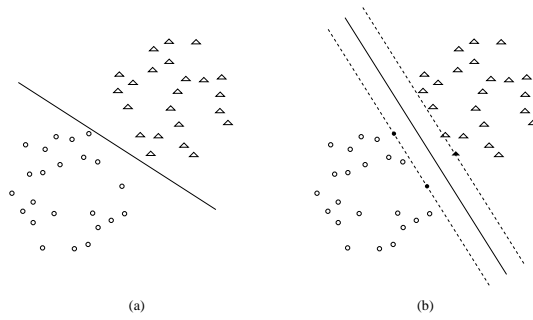


Figure 2: Separating hyperplane and optimal separating hyperplane. Both solid lines in (a) and (b) separate the two identical sets of open circles and triangles, but the solid line in (b) leaves the closest points (the filled circles and triangle) at the maximum distance. The dashed lines in (b) identify the margin.

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

The support vectors are the nearest points to the separating boundary. The intra-class distance, also called *margin*, is twice the distance of a support vector from the boundary, $2d$. The margin is an important quantity because it can be taken as an indicator of the separability of the data and consequently of the “goodness” of the representation used. In fact it can be theoretically shown that the quantity $\frac{R^2}{d^2}$, with R the radius of the smallest sphere containing the data points, is a measure of the model complexity of the SVM [34]. It may be that different input representations and/or kernel functions lead to quite different geometries of the data (i.e. different margins), which in turn influences the performance of the SVM.

2.2.1 A short overview of kernels

The central issue when using an SVM is the choice of the kernel function K . It is well known by now in the Kernel Machine community, that kernel K needs to satisfy some admissible properties known as Mercer property [34] which read: (i) K is continuous, (ii) $K(\mathbf{x}, \mathbf{t})$ is a symmetric function of \mathbf{x} and \mathbf{t} , and (iii) K is positive definite, i.e. for any integer m and any set of points $\{\mathbf{x}_1, \dots, \mathbf{x}_m\} \subset X$, matrix $M_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ is positive definite.

A consequence of (i)–(iii) is that $K(\mathbf{x}, \mathbf{t})$ can be re-written as

$$K(\mathbf{x}, \mathbf{t}) = \sum_{n=1}^{\infty} \lambda_n \phi_n(\mathbf{x}) \phi_n(\mathbf{t}). \quad (2)$$

where $\phi_n(\mathbf{x})$ are the eigenfunctions of the integral operator associated to kernel K and λ_n their eigenvalues [5]. By setting $\Phi(\mathbf{x})$ to be the sequence $(\sqrt{\lambda_n} \phi_n(\mathbf{x}))_n$, we see that K is a dot product in a Hilbert space of sequences (feature space), that is, $K(\mathbf{x}, \mathbf{t}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{t})$. Of course, kernel K can also be defined/built by choosing the ϕ 's and λ 's but, in general, those do not need to be known and can be implicitly defined by K .

Standard examples of Mercer kernels are the Gaussian kernel, $\exp \frac{\|\mathbf{x}-\mathbf{t}\|^2}{2\sigma^2}$, and an n^{th} degree polynomials $(\mathbf{x} \cdot \mathbf{y} + 1)^n$. Associated to kernel K there is also a space of functions where the SVM is learned/selected. Such a space is known as Reproducing Kernel Hilbert Space. See [34, 28, 6, 4] for more details on kernels. A rigorous mathematical development about kernels and RKHS is given in [5].

With a suitable choice of the kernel, the data can become separable in the feature space despite the fact that they may not be separable in the original input space. For example using a Gaussian kernel it is always possible to separate the data as soon as a sufficiently small

value of the variance of the Gaussian is chosen (likewise a polynomial kernel of sufficiently high degree can separate any data). If this parameter is too small or too large the SVM may overfit or underfit the data. When sufficient data is available this parameter can be found using a validation set. However it is also possible to get an estimate of the “best” kernel parameter(s) directly, without the use of validation data [3]

An important problem is that of devising methods building/engineering the kernel function K for a given machine learning problem. One approach is to choose a kernel which is invariant under certain transformations in space X which reflect the prior knowledge on the problem at hand [29]. A different approach consists of inferring a kernel from probabilistic models generating the data [12, 9]. We will come back to this issue in Section 5 where we will derive a kernel function from a probabilistic model generating images of an object class such as faces.

3 Comparison of representations for face and people detection

3.1 Pixels, principal components, and Haar wavelets

Using the experimental setup described above, we conducted experiments to compare the discriminative power of three different image representations:

- The *pixel representation*: train an SVM using the raw pixel values scaled between 0 and 1 (i.e. for faces this means that the inputs to the SVM machine are $19 \cdot 19 = 361$ dimensional vectors with values from 0 to 1 - before scaling it was 0 to 255).

- The *eigenvector (principal components) representation*: compute the correlation matrix of the positive examples (their pixel vectors) and find its eigenvectors. Then project the pixel vectors on the computed eigenvectors. We can either do a full rotation by taking the projections on all 361 eigenvectors, or use the projections on only the first few principal components. We discuss this issue in Section 3. We rescaled the projections to be between 0 and 1.
- The *wavelet representation*: consider a set of Haar wavelets at different scales and locations (see Figure 3), and compute the projections of the image on the chosen wavelets. For the face detection experiments we used all wavelets (horizontal, vertical and diagonal) at scales 4×4 and 2×2 since their dimensions correspond to typical features for the size of the face images considered. We had a total of 1,740 coefficients for each image. For the people detection system we considered wavelets at scales 32×32 and 16×16 shifted by 8 and 4 pixels respectively. We had a total of 1,326 coefficients. We rescaled the outputs of the projections to be between 0 and 1. A key motivation for using Haar wavelets comes from neuroscience, where there is a large body of literature (see for example [25]) supporting that neurons in the visual system, in particular at the early stages, act like oriented filters that have the shape of wavelets, many in the shape of oriented Haar wavelets. We have therefore tested this scheme in an engineering application.



Figure 3: The 3 types of 2-dimensional non-standard Haar wavelets; (a) “vertical”, (b) “horizontal”, (c) “diagonal”.

3.2 Experiments

Figure 4 shows the results of the experiments comparing the representations described above. In all these experiments a second order polynomial kernel was used. The motivation for using such a kernel is based on the experimental results of [21, 23]. Throughout the paper, notice the range of the axis in all the plots in the figures: the range varies in order to show clearer the important parts of the curves.

These experiments suggest a few remarks. First notice that both the pixel and eigenvector representations give almost identical results (small differences due to the way the ROC curves are produced are ignored). This is an observation that has a theoretical justification that we discuss in Section 3.3. Second, notice that for faces the wavelet representation performs about the same as the other two, but in the case of people, the wavelet representation is significantly better than the other two. This is a finding that was expected [23, 20]: for people pixels may not be very informative (i.e. people may have different color clothes), while wavelets capture intensity differences that discriminate people from other patterns [23]. On the other hand, for faces at the scale we used, pixel values seem to capture enough of the information that characterizes faces. Notice that all the three representations considered so far are linear transformations of the pixels representation. This takes us to the next topic.

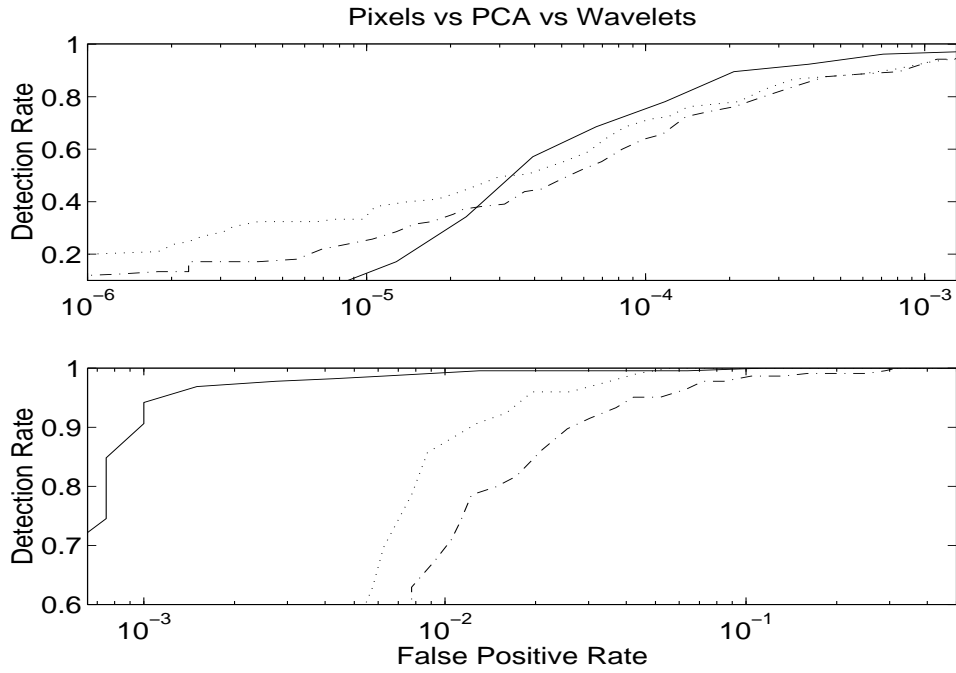


Figure 4: ROC curves for face (top) and people (bottom) detection: solid lines are for the wavelet representation, dashed lines for pixel representation, and dotted line for eigenvector representation (all 361 eigenvectors).

3.3 Linear transformations and kernels

As discussed in Section 2.2, a key issue when using a SVM is the choice of the kernel K in Equation (1). The kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ defines a dot product between the projections of two inputs $\mathbf{x}_i, \mathbf{x}_j$, in a feature space, noted with Φ in Section 2.2. Therefore the choice of the kernel is very much related to the choice of the “effective” image representation. So we can study different representations of the images through the study of different kernels for SVM.

In particular there is a simple relation between linear transformations of the original images, such as the ones considered above, and kernels. A point (image) \mathbf{x} is linearly decomposed in a set of features $\mathbf{c} = c_1, \dots, c_m$ by $\mathbf{c} = A\mathbf{x}$, with A a real matrix (we can think of the features \mathbf{c} as the result of applying a set of linear filters to the image \mathbf{x}). If the kernel used is a polynomial of degree m^4 (as in the experiments), then $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^m$, while $K(\mathbf{c}_i, \mathbf{c}_j) = (1 + \mathbf{c}_i \cdot \mathbf{c}_j)^m = (1 + \mathbf{x}_i^\top (A^\top A)\mathbf{x}_j)^m$. So using a polynomial kernel in the “ \mathbf{c} ” representation is the same as using a kernel $(1 + \mathbf{x}_i^\top (A^\top A)\mathbf{x}_j)^m$ in the original one. This implies that one can consider any linear transformation of the original images by choosing the appropriate square matrix $A^\top A$ in the kernel K of the SVM.

As a consequence of this observation, we have a theoretical justification of why the pixel and eigenvector representations lead to the same performance: in this case the matrix A is orthonormal, therefore $A^\top A = I$ which implies that the SVM finds the same solution in both cases. On the other hand, if we choose only some of the principal components (like in the case of eigenfaces [32]), or if we project the images onto a non-orthonormal set of Haar wavelets, the matrix A is no longer orthonormal, so the performance of the SVM may

⁴Generally this holds for any kernel for which only dot products between input arguments are needed - i.e. also for Radial Basis Functions.

be different. Figure 5 shows the kernel $K(\mathbf{x}, \mathbf{y})$ that results when the basis functions $\phi_i(\mathbf{x})$ in Equation (2) are a complete set of Haar wavelets, while the λ_i in (2) are decreasing as the frequency of the Haar wavelet $\phi_i(\mathbf{x})$ increases. Arguments \mathbf{x} and \mathbf{y} are 1-d in this case. Notice that if $\lambda_i = 1$ for all i , then $K(\mathbf{x}, \mathbf{y})$ would be 1 if $\mathbf{x} = \mathbf{y}$ and 0 otherwise, since a complete Haar basis is used as features.

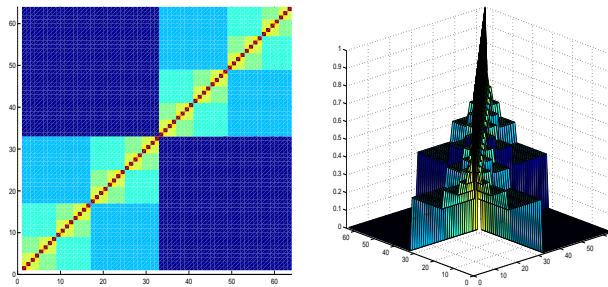


Figure 5: The kernel originating from Haar wavelet basis (see text) in a 1-d case. The figure shows two different views of the surface.

We can compare the performance of a machine that uses all eigenvectors (full rotation) or Haar wavelet projections with the one that uses only some of them - therefore matrix A above changes. This takes us to the following questions: first, how does SVM perform with high-dimensional input data? Second, how can we select some of the original input features in a principled way? These two questions are discussed in Section 4. Before discussing this, we briefly mention some experimental results regarding the effects of histogram equalization, a non-linear transformation, on the performance of the system.

3.4 Histogram equalization

We now discuss the experimental finding that histogram equalization (H.E.), a non-linear transformation of the “pixel” representation, improves the performance of the detection

system on our databases of images. Given an image, H.E. is performed in two steps: first the pixel values (numbering 0 to 255) are grouped into the smallest number of bins so that the distribution of the number of pixels in the image is uniform among the bins; then we replace the pixel values of the original image with the values (rank) of the bins they fall into. More information on H.E. can be found in the literature (i.e. [13]). Figure 6 shows an image of a face used for training, and the same face after H.E.

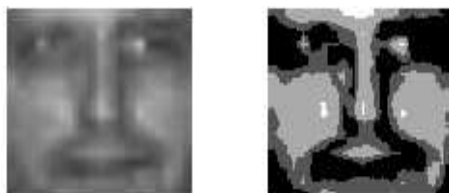


Figure 6: An original image of a face on the left. The same image after histogram equalization on the right.

We tested the systems described in Section 2.1, this time after performing H.E. on every input image. Only for the wavelet representation, instead of projecting histogram equalized images on the chosen wavelets, we transformed the outputs of the projections of the original images on the wavelets using a sigmoid function. This operation is (almost) equivalent to first performing H.E. and then projecting on the wavelet filters the histogram equalized image (assuming Gaussian-like histogram of the original image). Figure 7 shows the performance of the detection system. Both for face and people detection the performance increased dramatically.

H.E. has been extensively used for image compression and in this case it is straightforward to show that H.E. is a form of Vector Quantization [7] and is an effective coding scheme. Classification is however different from compression and it is an open question of why H.E. seems to improve so much the performance of our SVM classifier. Here we offer a conjecture:

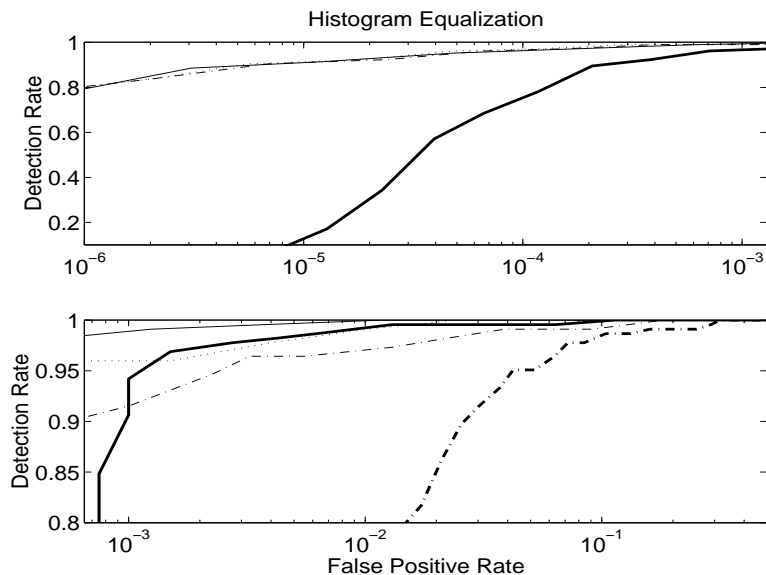


Figure 7: ROC curves for face (top) and people (bottom) detection after histogram equalization: solid lines are for the wavelet representation, dashed lines for the pixel representation, and dotted line for the eigenvector representation. We also show the ROC curve for the wavelet representation without histogram equalization (like in Figure 4); this is the bottom thick solid line. For people, the bottom thick dashed line shows the performance of pixels without H.E..

Suppose that a transformation satisfies the following two conditions:

- it is a legal transformation of the input vector, that is it preserves the class label;
- it increases the entropy of the input set, leading to a more compressed representation.

We *conjecture* that such a transformation will improve the performance of a SVM classifier.

Notice that H.E. is a transformation of the images that satisfies the above conditions: i.e. faces remain faces, and non-faces remain non-faces (of course one can design images where this does not hold, but such images are very unlikely and are not expected to exist among

the ones we used for training and/or testing). Moreover H.E. leads to a more compressed representation of the images (in terms of bits needed to describe them). Of course the first condition relies on prior information. In the case of H.E. applied to images we know a priori that H.E. is a transformation embedding "illumination invariance": images of the same face under different illuminations can be mapped into the same vector under H.E. Thus performing an H.E. transformation is roughly equivalent to using a larger training set containing many "virtual examples" generated from the real examples [8, 37] by changing the global illumination (mainly the dynamic range). Of course a larger training set in general improves the performance of a classifier. So this may explain the improvement of the system. The virtual examples (implicitly "added" by using a compressed representation through H.E.) capture partially the "a priori" information that changes in illumination do not affect the identity of the objects in the images.

In the case of the SVM classifier we have used, it is likely that H.E. makes the space of images "more discrete": there are fewer possible images. This may correspond to a better (more uniform) geometry of the training data (for example, after H.E. there may be a larger margin between the two classes) that leads to a better separating surface found by the SVM.

Thus our conjecture claims that H.E. improves classification performance because it does not change, say, faces into non-faces: this is "a priori" information about the illumination invariance of face images. H.E. exploits this information. One may be able to formalize this either through a compression argument, or through the equivalence with virtual face examples, or possibly through a geometric argument. This "a priori" information is true for face and people images, but may not hold for other ones, in which case H.E. might not improve performance. In general, because of the same arguments outlined above, we

expect that any transformation of the original images that “effectively” takes advantage of prior information about the class of images considered and compresses their signatures, will improve the performance of the system.

4 Input feature selection using SVM

Multimedia data, such as images or video, are typically represented or stored as very high dimensional vectors. For example if one chooses to represent images using the pixel representations, then for example a 100×100 image is a vector of 10000 numbers in the black and white case (30000 for the a standard color case such as RGB). Clearly the processing time for searching or performing other operations for such systems is highly impacted by the fact that the data are so high dimensional. It is therefore practically important to find compact representations of multimedia data, while at the same time not affecting significantly the performance of systems such as detection/search ones (due to loss of information because of the compactness of the representation).

This section addresses the issue of selecting only a few of the features typically used for representing images without degrading the performance of our detection system. One important problem with features selection is the multiple use of the data: the same training set is used first to train the system using all the features, then to select the important features, and finally to retrain the system using only the selected features. The multiple use of training data may lead to overfitting, so it is unclear a priori that selecting features can improve performance.

In order to investigate these issues, we performed several experiments where the object

detection systems were trained with different numbers of input features. To this purpose we have developed a method for automatically selecting a subset of the input features within the framework of SVM.

4.1 A Method for feature selection using SVM

Our feature selection method is based on the observation that the most important input features are the ones for which, when removed or modified, the separating boundary $f(\mathbf{x}) = 0$ varies the most. Instead of the variation of the boundary we can consider the average variation of the value of the function $f(\mathbf{x})$ in a region around the separating boundary. The motivation is that perturbations of the separating boundary ($f(\mathbf{x}) = 0$) will influence the classification of points only in the area around the separating boundary, and in classification we are interested not in the actual values of the function $f(\mathbf{x})$ but in the classification of the points (based on the sign of f). For points away from the separating boundary, any changes to the real values of f are not important as long as the sign of f in those areas does not change, which we can safely assume to be the case if f is only perturbed and therefore a very positive value still remains positive and a very negative one is still negative.

We approximate the average change of the value of the function $f(\mathbf{x})$ when we remove a feature with the average change of f when we perturb the values of a feature. Since we are interested in the relative importance of the features, we make the simplifying assumption that if the average change of the values of f around the separating boundary when we remove feature r is bigger than that when we remove feature t , then the average change of f when we perturb the values of feature r is also bigger than that when we perturb the values of

feature t . Under this assumption, we estimate the average change of f in the neighborhood around the separating boundary $f(\mathbf{x}) = 0$ by computing the derivative of $f(\mathbf{x})$ with respect to an input feature x^r and integrating the absolute value (we are interested in the magnitude of the derivative) in a volume V around the boundary:

$$I^r = \int_V dP(\mathbf{x}) \left| \frac{df}{dx^r} \right|.$$

In practice we cannot compute this quantity because we do not know the probability distribution $P(\mathbf{x})$ of the data (the images). Instead we can approximate I_r with the sum over the support vectors⁵:

$$I^r \approx \sum_{i=1}^{N_{sv}} \left| \frac{df}{dx_i^r} \right| = \sum_{i=1}^{N_{sv}} \left| \sum_{j=1}^{N_{sv}} \alpha_j y_j K^r(\mathbf{x}_j, \mathbf{x}_i) \right|. \quad (3)$$

where N_{sv} is the number of support vectors and $K^r(\mathbf{x}_j, \mathbf{x}_i)$ is the derivative of the kernel with respect to the r^{th} dimension evaluated at \mathbf{x}_i . For example for $K(\mathbf{x}_j, \mathbf{x}_i) = (1 + \mathbf{x}_j \cdot \mathbf{x}_i)^2$, this is equal to $K^r(\mathbf{x}_j, \mathbf{x}_i) = (1 + \mathbf{x}_j \cdot \mathbf{x}_i) \mathbf{x}_i^r$ where \mathbf{x}_i^r is the r^{th} component of vector \mathbf{x}_i .

Once the I_r 's in Equation (3) have been computed, we can rank them in decreasing order and select the top s features according to this rank. Alternatively we can fix a parameter $g \in (0, 1)$ and select those feature for which I_r is greater then $g(I_{\max} - I_{\min}) + I_{\min}$. Below we follow the first approach.

Notice that the computation of the derivative in Equation (3) is only an approximation to the actual derivative: changing the value of a feature may also lead to different solution of the SVM, namely different α 's. We assume that this change is small and we neglect it. To

⁵For separable data these are also the points nearest to the separating surface. For non-separable data we can take the sum over only the support vectors near the boundary.

optimized this, we can choose large s (i.e. only few features are discarded), and iterated the feature selection procedure. However, in the experiment below we have found that selecting the feature in one step gives very similar results to an iterative procedure.

4.2 Experiments

For people detection, using the proposed method we selected 29 of the initial set of 1,326 wavelet coefficients. We then trained an SVM using only the 29 selected features and compared the performance of the machine with that of an SVM trained on 29 coefficients selected using a manual method as described in [23]. We show the results in Figure 8 (bottom plot).

We also tested the same method for face detection. We selected 30 of the initial 1,740 wavelet coefficients, and we compared the performance of an SVM trained using only these 30 features, with the performance of a SVM that uses 30 randomly selected features out of the initial 1,740. We also show the performance of the system when 500 of the wavelets were chosen. Notice that using the proposed method we can select about a third (500) of the original input dimensions without significantly decreasing the performance of the system. The result is also shown in Figure 8 (top plot). Finally for the eigenvector representation, we also tested the system using few principal components (so we did not use our feature selection method for this case). The results are also shown in Figure 8 (middle plot).

From all the experiments shown in Figure 8 we can make two main observations: first notice that our feature selection method performs significantly better than manual feature selection in the case of pedestrian detection, and also significantly better than random feature selection in the case of face detection. This is an indication that our method is performing

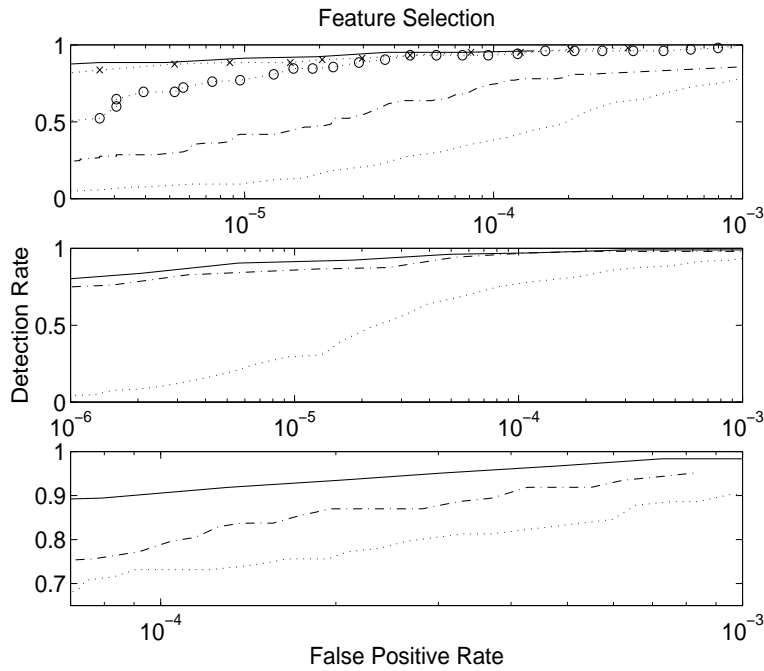


Figure 8: *Top figure:* solid line is face detection with all 1,74 wavelets, dashed line is with 30 wavelets chosen using the proposed method, and dotted line is with 30 randomly chosen wavelets. The line with \times 's is with 500 wavelets, and the line with \circ 's is with 120 wavelets, both chosen using the method based on Equation (3). *Middle figure:* solid line is face detection with all eigenvectors, dashed line is with the 40 principal components, and dotted line is with the 15 principal components. *Bottom figure:* solid line is people detection using all 1,326 wavelets, dashed line is with the 29 wavelets chosen by the method based on Equation 3, and dotted line is with the 29 wavelets chosen in [23]

experimentally well. The second interesting observation is that in general SVM are not sensitive to large numbers of input dimensions. In fact in all cases, when using all input dimensions (all wavelets or all eigenvectors) the system performed better (or about the same) than when using few of the input features. Our experimental finding indicates that SVM work well even when the input data are high-dimensional, by automatically dealing with irrelevant features. Of course, as discussed above, for multimedia databases it is often

important to find compact representations for computational purposes, so to that end our feature selection method can be used without significantly degrading the performance of multimedia analysis systems.

5 Features from probabilistic models

In this section we take a different approach to the problem of finding image representations. Consider a specific class of images (e.g. faces or people) and assume that they are sampled according to a generative probabilistic model $P(\mathbf{x}|\beta)$, where β indicates a set of parameters. As an example consider a Gaussian distribution:

$$P(\mathbf{x}|\beta) = \frac{1}{2\pi^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} \exp \left\{ -\frac{1}{2}(\mathbf{x} - \mathbf{x}_0)^\top \Sigma^{-1}(\mathbf{x} - \mathbf{x}_0) \right\} \quad (4)$$

where the parameters β are the average image \mathbf{x}_0 and the covariance Σ .

Recent work [22] shows how the assumption of a specific probabilistic model of the form (4) suggests a choice of the kernel – and therefore of the “features” – to be used in order to reconstruct, through the SVM regression algorithm, images of faces and people. The relevant features are the principal components \mathbf{u}_n of the set of examples (i.e. faces or people) scaled by the corresponding eigenvalues λ_n . However, [22] left open the question of what features to choose in order to do classification as opposed to regression, that to discriminate faces (or people) from non-faces (non-people), once the probabilistic model is decided.

Very recently a general approach to the problem of constructing features for classification

based on a probabilistic models describing the training examples has been suggested [12]. The choice of the features was made implicitly through the choice of the kernel to be used for a kernel classifier. In [12] a probabilistic model for both the classes to be discriminated was assumed, and the results were also used when a model of only one class was available - which is the case we have.

Let us denote with $L(\mathbf{x}|\beta)$ the log of the probability function and define the Fisher information matrix

$$I = \int d\mathbf{x} P(\mathbf{x}|\beta) \partial_i L(\mathbf{x}|\beta) \partial_j L(\mathbf{x}|\beta),$$

where ∂_i indicates the derivative with respect to the parameter β_i . A natural set of features, ϕ_i , is found by taking the gradient of L with respect to the set of parameters,

$$\phi_i(\mathbf{x}) = I^{-\frac{1}{2}} \frac{\partial L(\mathbf{x}|\beta)}{\partial \beta}. \quad (5)$$

These features were theoretical motivated in [12]. The basic idea is that, when $P(\mathbf{x}|\beta)$ is an exponential family, the gradient of the likelihood function L above w.r.t. the model parameters essentially contains a sufficient statistics for the example \mathbf{x} . Thus, such features captures all the information contained in the model for classification. In [12] is also shown that the features in (5) or, equivalently, the kernel in Equation 6 below, lead to kernel classifiers which are asymptotically as least as discriminative as the Bayes classifier based on the given generative model.

We have assumed the generative model (4) and rewritten it with respect to the average image \mathbf{x}_0 and the eigenvalues λ_n and obtain the set of features according to Equation (5).

For simplicity the principal components were kept fixed in the model. The features obtained in this way were then used as a new input representation in our learning system. The resulting kernel obtained by taking the dot product between the features (dot product for the implicitly chosen representation) of a pair of images \mathbf{x}_i and \mathbf{x}_j is:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{n=1}^N [-\lambda_n^{-1}(c_n(\mathbf{x}_i) - c_n(\mathbf{x}_j))^2 + \lambda_n^2 c_n(\mathbf{x}_i) c_n(\mathbf{x}_j)], \quad (6)$$

where $c_n(x) = (\mathbf{x} - \mathbf{x}_0)^\top \mathbf{u}_n$ and N is the total number of eigenvectors (principal components) used. The parameters \mathbf{x}_0 , \mathbf{u}_n , λ_n were estimated using the training examples of faces (or people). Note that we used the training data multiple times: once for estimating the parameters of the probabilistic model (4), and once to train an SVM classifier.

Notice also that the new features are a non-linear transformation of the pixel representation and the eigenvalues appear in the denominator of each term in the kernel. This is not surprising as the smaller principal components may be important for discrimination: for example, in the problem of face recognition we expect that to discriminate between two faces, we can get more benefit by looking at small details in the image which may not be captured by the larger principal components. Similarly, in the problem of face detection, the non-image class is expected to have the same energy on each principal component, so the small principal components may still be useful for classification. On the other hand, when the goal is to reconstruct or de-noise the *in-class images*, we deal with a regression-type problem; in such a case the top principal components capture the most important coefficients for reconstructing the images and only few of them need to be considered.

Equation (6) indicates that only a limited number of principal components can be used

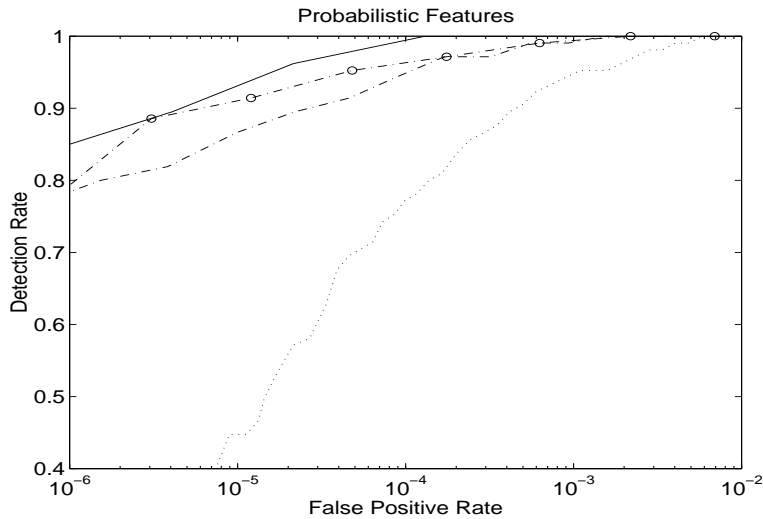


Figure 9: Face experiments: Solid line indicates the probabilistic features using 100 principal components, dashed line is for 30 principal components, and dotted for 15. We also show the ROC curves with all wavelets (line with circles) for comparison. Histogram equalization was performed on the images.

in practice because small λ_n create numerical instabilities in the learning algorithm. We performed several experiments by changing the number of principal components used in the model (see Figure 9) and compared the results with the image representations discussed in Section 3. We notice that the proposed representation performs slightly better than the other ones (when 100 principal components were used), but not significantly better. It may be the case that features from other (more realistic) probabilistic models lead to better systems.

6 Conclusions

In this paper we have explored the important multimedia information systems issue of choosing “good” representations for multimedia data. We focused on image data, and on a particular data processing task, namely that of detecting objects in images for example for the

purpose of searching in an intelligent multimedia database.

The possible multimedia representations are of course endless. In this paper we have presented a number of experiments for face and people detection with different image representations and kernels, from the standard pixel based ones to more advanced new ones, using SVM as the central machine learning method. Our experiments indicate that the representation chosen depends, as expected, on the type of images analyzed. We showed for example that for images of pedestrians the use of Haar wavelets lead to significantly better analysis performance than the use of pixels or principal components, while for images of faces all these three representations lead to similar results. We also presented a method for choosing features that can be used for finding compact multimedia representations without significant degradation of the performance of the data analysis systems. Finally we explored new directions for finding image/multimedia representations, namely through the use of features generated from probabilistic models of images, and conjectured a framework for choosing representations that can lead to better multimedia analysis systems that we developed through the study of the effects of histogram equalization.

Acknowledgments

Most of this work was done while the authors were with the Center for Biological and Computational Learning at MIT. We wish to thank Tommi Jaakkola and Marina Meila for helpful discussions on generative models.

References

- [1] N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *Symposium on Foundations of Computer Science*, 1993.
- [2] M. Betke and N. Makris. Fast object recognition in noisy images using simulated annealing. In *Proceedings of the Fifth International Conference on Computer Vision*, pages 523–20, 1995.
- [3] O. Chapelle and V. Vapnik. Model selection for support vector machines. In *Advances in Neural Information Processing Systems*, 1999.
- [4] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines*. Cambridge University Press, 2000.
- [5] F. Cucker and S. Smale. On the mathematical foundations of learning. Preprint, 2001.
- [6] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.
- [7] A. Gersho and R.M. Gray. *Vector quantization and signal compression*. Kluwer Academic Publishers, Boston, 1991.
- [8] F. Girosi and N. Chan. Prior knowledge and the creation of “virtual” examples for RBF networks. In *Neural networks for signal processing, Proceedings of the 1995 IEEE-SP Workshop*, pages 201–210, New York, 1995. IEEE Signal Processing Society.
- [9] D. Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, University of California, Santa Cruz, 1999.
- [10] B. Heisele, U. Kressel, and W. Ritter. Tracking Non-rigid, Moving Objects Based on Color Cluster Flow. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, 1997.

- [11] D. Hogg. Model-based vision: a program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.
- [12] T. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proc. of Neural Information Processing Conference*, 1998.
- [13] Anil K. Jain. *Fundamentals of digital image processing*. Prentice-Hall Information and System Sciences Series, New Jersey, 1989.
- [14] T. Joachims. Text categorization with support vector machines. Technical Report LS-8 Report 23, University of Dortmund, November 1997.
- [15] M. Kearns and R.E. Shapire. Efficient distribution-free learning of probabilistic concepts. *Journal of Computer and Systems Sciences*, 48(3):464–497, 1994.
- [16] M.K. Leung and Y-H. Yang. A region based approach for human body analysis. *Pattern Recognition*, 20(3):321–39, 1987.
- [17] S. McKenna and S. Gong. Non-intrusive person authentication for access control by visual tracking and face recognition. In J. Bigun, G. Chollet, and G Borgefors, editors, *Audio- and Video-based Biometric Person Authentication*, pages 177–183. IAPR, Springer, 1997.
- [18] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. Technical Report 326, mitmedia, 1995.
- [19] P. Niyogi, C. Burges, P. Ramesh. Distinctive feature detection using support vector machines. In *Proc. of Int. Conf. on Acoustic, Speech, and Signal Processing*, Phoenix, Arizona, 1999.
- [20] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *Proc. Computer Vision and Pattern Recognition*, pages 193–199, Puerto Rico, June 16–20 1997.

- [21] E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In *IEEE Workshop on Neural Networks and Signal Processing*, Amelia Island, FL, September 1997.
- [22] C. Papageorgiou, F. Girosi, and T. Poggio. Sparse correlation kernel based signal reconstruction. Technical Report 1635, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1998. (CBCL Memo 162).
- [23] C. Papageorgiou, M. Oren, and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38, 1, 15-33, 2000.
- [24] M. Pontil, and A. Verri. Object Recognition with Support Vector Machines. *IEEE Trans. on Pattern Anal. and Mach. Intell.*, Vol. 20, 637-646, 1998.
- [25] M. Riesenhuber, and T. Poggio. Models of Object Recognition *Nature Neuroscience*, 3 Supp., 1199-1204, 2000
- [26] K. Rohr. Incremental recognition of pedestrians from image sequences. *Computer Vision and Pattern Recognition*, pages 8–13, 1993.
- [27] H. Rowley, S. Baluja, and T. Kanade. Human Face Detection in Visual Scenes. Technical Report 95–158, CMU CS, July 1995. Also in *Advances in Neural Information Processing Systems* (8):875-881.
- [28] B. Scholkopf, C. Burges, and A. Smola. *Advances in kernel methods – Support vector learning*. MIT Press, 1998.
- [29] B. Scholkopf, P. Simard, A. Smola, and V. Vapnik. Prior knowledge in support vector kernels. In *Advances in Neural Information Processing Systems 9*, 1997.

- [30] 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.
- [31] T. Tsukiyama and Y. Shirai. Detection of the movements of persons from a sparse sequence of tv images. *Pattern Recognition*, 18(3/4):207–13, 1985.
- [32] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proceedings CVPR*, pages 586–591, Hawaii, June 1991.
- [33] R. Vaillant, C. Monrocq, and Y. Le Cun. Original approach for the localisation of objects in images. *IEEE Proc. Vis. Image Signal Process.*, 141(4), August 1994.
- [34] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.
- [35] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Th. Prob. and its Applications*, 17(2):264–280, 1971.
- [36] R. C. Veltkamp and M. Tanase. Content-based image retrieval systems: A survey Tech. Report UU-CS-2000-34, Dept. of Computing Science, Utrecht Univ., Netherlands, 2000.
- [37] T. Vetter, T. Poggio, and H. Bülthoff. The importance of symmetry and virtual views in three-dimensional object recognition. *Current Biology*, 4(1):18–23, 1994.
- [38] G. Wahba. *Splines Models for Observational Data*. Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.
- [39] A. Yuille, P. Hallinan, and D. Cohen. Feature Extraction from Faces using Deformable Templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.
- [40] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. Technical Report 353, MIT Media Laboratory, 1995.

Theodoros Evgeniou is an Assistant Professor of Information Systems in the Technology Management department of INSEAD. He holds two BS, an M.Eng., and a PhD in Computer Science from MIT, where he has been affiliated with the Artificial Intelligence Laboratory and the Center for Biological and Computational Learning. His current research interests are in the areas of data mining and business intelligence.

Massimiliano Pontil received his Bachelor and Ph.D. in Theoretical Physics from the University of Genova in 1994 and 1999. Since October 2000 he has been with the Department of Information Engineering, University of Siena, Italy. His research interests involve machine learning, pattern recognition, and statistics. Previously, he has been a visiting student at MIT in 1998, a Post-doctoral Fellow in 1999 and 2000 at the Center for Biological and Computational Learning at MIT and a Research Fellow at City University of Hong Kong in 2001. He has also spent some time as a visiting researcher at the RIKEN Brain Science Institute, Tokyo, and AT&T Labs, USA. He has published about 30 papers in international journals and conferences on different aspects of learning theory, machine learning, and computer vision.

Constantine Papageorgiou received his Ph.D. in Computer Science at MIT in 1999 with a specialization in artificial intelligence, pattern recognition, machine learning, and natural language processing. The system for trainable object detection in images that he developed as part of his thesis was deployed in a prototype DaimlerChrysler automobile. His industry experience includes BBN Systems and Technologies, Kana Communications, and iSpheres Corporation. In addition, he has consulted for startups in several areas, including wireless location technologies, intelligent agents, and aesthetic profiling.

Tomaso Poggio, Ph.D. is an Uncas and Helen Whitaker Professor of Vision Sciences and

Biophysics, in the Department of Brain and Cognitive Sciences; Co-Director, Center for Biological and Computational Learning; Member for the last 20 years of the Artificial Intelligence Laboratory at MIT; and, since 2000, member of the faculty of the McGovern Institute for Brain Research. Earlier Prof. Poggio had worked on the visual system of the fly with W. Reichardt in Tuebingen at the Max Planck Institut fuer Biologische Kybernetik and with D. Marr on computational analysis of human and machine vision. He was responsible for the Vision Machine project at the AI Lab. Serving on the editorial boards of a number of leading interdisciplinary journals, Professor Poggio is a Founding Fellow of the American Association for Artificial Intelligence, an Honorary Associate of the Neuroscience Research Program at Rockefeller University and a member of several scientific and engineering associations including IEEE, AAAS; he is an elected member of the American Academy of Arts and Sciences. Professor Poggio received his doctorate in theoretical physics from the University of Genoa in 1970, had a tenured research position at the Max Planck Institute from 1971 to 1981 when he became Professor at MIT. He is the author of hundreds of papers in areas ranging from biophysics to information processing in man and machine, artificial intelligence, machine vision and learning. A former Corporate Fellow of Thinking Machines Corporation, he was and is still peripherally involved in several companies in the areas of bioinformatics, computer graphics, computer vision, computer networks, and financial engineering.