Machine Learning Strategies for Complex Tasks

Colin Campbell[1], Theodoros Evgeniou[2], Bernd Heisele[2], and Massimiliano Pontil[2]

[1] Department of Engineering Mathematics,
Bristol University, Bristol BS8 1TR,
United Kingdom
C.Campbell@bristol.ac.uk
[2] Center for Biological and Computational Learning,
MIT, Cambridge MA 02142,
USA
{theos,pontil,heisele}@ai.mit.edu

Abstract. In this paper we begin by reviewing recent research on kernel methods. This subject provides a systematic and principled approach to machine learning tasks such as classification, regression, novelty detection, and query learning. Advanced robots are examples of complex autonomous systems which must be able to complete sophisticated operations involving one or more of these tasks. For example, regression is relevant to modeling the coordinate transformations of manipulator kinematics. Analysis of images in a scene may require detection of novel objects and classification of known objects. We outline past work illustrating successful application of kernel methods to object recognition in scenes. The complex machine vision and control operations inherent in humanoid robotics suggests the area is an excellent test-bed for co-operatively integrating different machine learning tasks and stimulating future research directions.

1 Introduction

In this paper we will begin by outlining a new approach to machine intelligence based on *kernel methods*. This approach is systematic and properly motivated theoretically [64]. Training consists of the minimization of a convex cost function, so there is only one solution. There are also few tunable parameters to adjust and no need for a lot of experimentation to determine the model's architecture, unlike neural networks, for example. Most importantly kernel methods perform well in practice and exhibit good generalization on real-life datasets.

The purpose of writing this paper is threefold. Firstly, to introduce kernel methods to a wider audience. Secondly, to outline their successful application to important tasks in humanoid robotics such as 3D object recognition and obstacle detection [48, 16, 43]. Finally we discuss future perspectives and the development of more complex intelligent systems. Many machine learning tasks such as classification, regression and novelty detection are now reasonably well understood. Thus to progress machine intelligence it is interesting to consider more complex vision or robotic systems in which these tasks are only sub-components. For example, in the analysis of a scene objects may be novel or known. Known objects may be classified and an appropriate response generated. Novel objects will lead to learning and possible queries to extract additional information.

The structure of the paper is as follows. In section 2 we will introduce kernel methods and outline their use for binary and multi-class classification, regression, novelty detection and query learning. In section 3 we will then illustrate their application to machine vision. Finally, we discuss future perspectives. For the sake of brevity we will concentrate on Support Vector Machines (SVMs) which are the most well-known approach based on kernel methods.

2 An Introduction to Kernel Methods.

2.1 A Unified View of the Learning Methods

We will consider learning techniques which lead to solution of the form

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i K(\mathbf{x}, \mathbf{x}_i). \tag{1}$$

where the $\mathbf{x}_i, i = 1, \ldots, m$ are the input examples, K a certain symmetric positive definite function named kernel (see below), and α_i a set of parameters to be determined form the examples. For all the machines considered, the

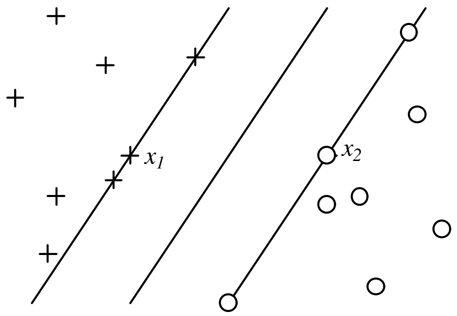


Fig. 1. The *margin* is the perpendicular distance between the separating hyperplane and a hyperplane through the closest points (these are *support vectors*). The region between the hyperplanes on each side is called the *margin band.* $\mathbf{x}_1$ and $\mathbf{x}_2$ are examples of support vectors of opposite sign.

function f is found by minimizing functionals of the type

$$H[f] = ||f||_K^2 \; + \; C \, \sum_{i=1}^{m} V(y_i, f(\mathbf{x}_i)), \tag{2}$$

where V is a *loss function* which measures the goodness of the predicted output $f(\mathbf{x}_i)$ with respect to the given output y_i (for the case of supervised learning - for other cases, such as the novelty detection one discussed below, there is no output y_i), $||f||_K^2$ a smoothness term which can be thought of as a norm in the Reproducing Kernel Hilbert Space defined by the kernel K, and C a positive parameter which controls the relative weight between the data and the smoothness term (see below). The choice of the loss function determines different learning techniques, each leading to a different learning algorithm for computing the coefficients α_i [17]. A large family of these machines has been analyzed and justified theoretically [17]. The main technique we discuss in this paper is Support Vector Machines (SVM) [12, 64, 52].

2.2 Binary Classification.

To introduce SVMs we will start with the simplest case of binary classification. Theoretical results have been derived which bound the generalization error [63, 64, 13, 54] for binary classification (that is, the probability of misclassifying a future point). In particular, these theoretical bounds have two implications. Firstly, the generalization error bound is minimized by maximizing the minimal distance between the hyperplane separating the two classes and the closest datapoints to the hyperplane (Figure 1). This minimal distance will be called the *margin* and denoted γ. The second observation is that the generalization error bound does not depend on the dimension of the space. This motivates the idea of *kernel substitution* which amounts to a nonlinear projection of data into a high-dimensional space where it is easier to separate the two classes of data.

The Learning Task. Let us consider a binary classification task with datapoints $\mathbf{x}_i$ $(i = 1, \ldots, m)$ having corresponding labels $y_i = \pm 1$ and let us suppose the decision function is:

$$f(\mathbf{x}) = \mathrm{sign}\,(\mathbf{w} \cdot \mathbf{x} + b) \tag{3}$$

If the dataset is separable then the data will be correctly classified if $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) > 0 \; \forall i$. Clearly this relation is invariant under a positive rescaling of the argument inside the $sign$-function, hence we can define a *canonical hyperplane* such that $\mathbf{w} \cdot \mathbf{x} + b = 1$ for the closest points on one side and $\mathbf{w} \cdot \mathbf{x} + b = -1$ for the closest on the other. For the separating hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$ the normal vector is clearly $\mathbf{w} / \, ||\mathbf{w}||$. Hence the margin is given by the projection of $\mathbf{x}_1 - \mathbf{x}_2$ onto this vector where $\mathbf{x}_1$ and $\mathbf{x}_2$ are the closest points on opposite sides of the separating hyperplane (see Figure 1). Since $\mathbf{w} \cdot \mathbf{x}_1 + b = 1$ and $\mathbf{w} \cdot \mathbf{x}_2 + b = -1$ this means the margin is $\gamma = 1 / \, ||\mathbf{w}||$. To maximize the margin the task is therefore:

$$\min \left[\frac{1}{2} \, ||\mathbf{w}||^2 \right] \tag{4}$$

subject to the constraints:

$$y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b \right) \geq 1 \qquad \forall i \tag{5}$$

Taking the Wolfe dual, we can solve for the dual parameters (Lagrange multipliers) α_i by maximizing with respect to the α_i the dual cost function [40, 12]:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_i \cdot \mathbf{x}_j \right) \tag{6}$$

subject to the constraint:

$$\alpha_i \geq 0 \qquad \sum_{i=1}^{m} \alpha_i y_i = 0 \tag{7}$$

and the primal solution $\mathbf{w}$ can be shown [40, 12] to be given by

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i. \tag{8}$$

Kernel substitution. This constrained quadratic programming (QP) problem will give an optimal separating hyperplane which maximizes the margin if the data is separable. However, we have still not exploited the second observation made earlier: namely, the error bound does not depend on the dimension of the space. This feature enables us to give an alternative kernel representation of the data which is equivalent to a mapping into a high dimensional space (called *feature space*) where the two classes of data are more readily separable. For the dual objective function in (6) we notice that the datapoints, $\mathbf{x}_i$, only appear inside an inner product. Thus this mapping is achieved through a replacement of the inner product:

$$\mathbf{x}_i \cdot \mathbf{x}_j \rightarrow \phi\left(\mathbf{x}_i\right) \cdot \phi(\mathbf{x}_j) \tag{9}$$

The functional form of the mapping $\phi(\mathbf{x}_i)$ *does not need to be known* since it is implicitly defined by the choice of *kernel:*

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \tag{10}$$

which is the inner product in the higher dimensional feature space (feature space must therefore be a Hilbert or inner product space [3, 67, 64]). With a suitable choice of kernel the data can become separable in feature space despite the fact that it may not be separable by a hyperplane in the original input space. A number of choices can be made [3, 67, 64] for the kernel function, for example:

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma^2} \tag{11}$$

defining an RBF network and:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^d \qquad K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta \mathbf{x}_i \cdot \mathbf{x}_j + b) \tag{12}$$

which would define polynomial and feedforward neural network classifiers. Each choice of kernel will define a different type of feature space and the resulting classifiers will perform differently on test data though the generalization bounds mentioned earlier imply good performance on new data.

For the given choice of kernel the learning task therefore involves maximization of the objective function:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{13}$$

subject to the constraints (7) and test examples are evaluated using a decision function given by the sign of:

$$f(\mathbf{z}) = \sum_{i=1}^{m} y_i \alpha_i K(\mathbf{x}_i, \mathbf{z}) + b \tag{14}$$

Since the bias, b, does not feature in the above dual formulation it is found from the primal constraints via:

$$b = -\frac{1}{2}\left[\max_{\{i|y_i=-1\}}\left(\sum_{j\in\{\mathrm{SV}\}}^{m} y_j\alpha_j K(\mathbf{x}_i,\mathbf{x}_j)\right) + \min_{\{i|y_i=+1\}}\left(\sum_{j\in\{\mathrm{SV}\}}^{m} y_j\alpha_j K(\mathbf{x}_i,\mathbf{x}_j)\right)\right] \tag{15}$$

using the optimal values of α_j. When the maximal margin hyperplane is found in feature space, only those points which lie closest to the hyperplane have $\alpha_i > 0$ and these points are the *support vectors* (SV) [12, 64]. All other points have $\alpha_i = 0$ and correspond to non-support vectors. These datapoints do not influence the position and orientation of the separating hyperplane and hence do not contribute to the hypothesis (Figure 1). This means that the representation of the hypothesis is given solely by those points which are closest to the hyperplane which are thus the most informative patterns in the data.

Allowing for Training Errors: Soft Margin Techniques. Most real life datasets contain noise and an SVM can fit this noise leading to poor generalization. The effect of noise can be reduced by introducing *soft margin* [12] training errors ξ_i and a trade off between training error and margin decided by parameter C as in (2). In this case the primal formulation of SVM becomes:

$$\min\left[\frac{1}{2}\left|\left|\mathbf{w}\right|\right|^2\right] + C\sum_{i=1}^{m}\xi_i \tag{16}$$

subject to the constraints:

$$y_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) \geq 1 - \xi_i \qquad \forall i \tag{17}$$

$$\xi \geq 0 \qquad \forall i \tag{18}$$

This can be seen as a machine of the form (2) [17]. Taking the dual of this problem we end up with a maximization which is the same as before except for the introduction of the following constraint on the α_i parameters [12]:

$$0 \leq \alpha_i \leq C \tag{19}$$

The final solution is again of the form

$$f(\mathbf{z}) = \sum_{i=1}^{m} y_i\alpha_i K(\mathbf{x}_i,\mathbf{z}) + b \tag{20}$$

which is of the form (1).

Determining the kernel parameters. During the training process the kernel parameter needs to be specified (e.g. σ for the RBF kernel). If this parameter is too small or too large the model may overfit or underfit the data. If sufficient data is available this parameter can be found using validation data by testing performance against different choices of kernel parameter. However, it is also possible to get an estimate of the kernel parameter more directly without use of validation data [11].

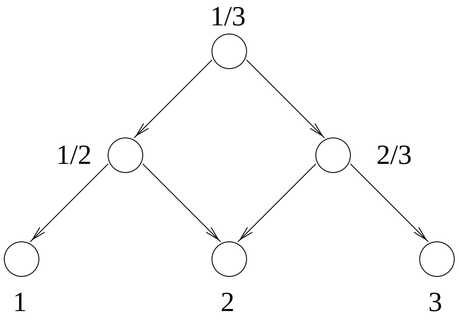
Multiclass Classification. Various schemes have been proposed to handle multiclass classification [33, 69]. One of the simplest schemes [46] is to reduce multiclass classification to a series of binary classification operations at each node in the tree (Figure 2). Both bottom up [48] and top-down trees [46] can be used. We discuss the first case in section 3.1.

2.3 Novelty Detection.

In many vision applications it is important to distinguish novel objects from known objects. One approach to novelty detection is to create a binary-valued function which is positive in those regions of input space where data predominantly lies and negative elsewhere.

One approach [58] is to find a hypersphere with a minimal radius R and center $\mathbf{a}$ which contains most of the data: novel test points lie outside the boundary of this hypersphere. The technique we now outline was originally suggested by Vapnik [63, 9], interpreted as a novelty detector by Tax and Duin [58] and used by the latter authors for real life applications [58]. The effect of outliers is reduced by using slack variables ξ_i to allow for datapoints outside the sphere and the task is to minimize the volume of the sphere and number of datapoints outside i.e.

$$\min\left[R^2 + \ C\ \sum_{i=1}^{m}\xi_i\right]$$



g. 2. A multi-class classification problem can be reduced to a series of binary classification tasks using a tree structure with nary decision at each node. A 3-class (1,2,3) case is shown here.

ıbject to the constraints:

$$(\mathbf{x}_i - \mathbf{a})^T(\mathbf{x}_i - \mathbf{a}) \leq R^2 + \xi_i$$

and $\xi_i \geq 0$, and where C controls the tradeoff between the two terms.

After kernel substitution the dual formulation amounts to maximisation of:

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{m} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{21}$$

with respect to α_i and subject to $\sum_{i=1}^{m} \alpha_i = 1$ and $0 \leq \alpha_i \leq C$. If $C < 1$ then *at bound* examples will occur with $\alpha_i = C$ and these correspond to outliers in the training process. Having completed the training process a test point $\mathbf{z}$ is declared novel if:

$$K(\mathbf{z}, \mathbf{z}) - 2\sum_{i=1}^{m} \alpha_i K(\mathbf{z}, \mathbf{x}_i) + \sum_{i,j=1}^{m} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) - R^2 \geq 0 \tag{22}$$

where R^2 is first computed by finding an example which is *non-bound* and setting this inequality to an equality.

An alternative approach has been developed by Schölkopf et al. [53]. Suppose we restrict our attention to RBF kernels: in this case the data lie in a region on the surface of a hypersphere in feature space since $\phi(\mathbf{x}) \cdot \phi(\mathbf{x}) = K(\mathbf{x}, \mathbf{x}) = 1$. The objective is therefore to separate off this region from the surface region containing no data. This is achieved by constructing a hyperplane which is maximally distant from the origin with all datapoints lying on the opposite side from the origin and such that $\mathbf{w} \cdot \mathbf{x}_i + b \geq 0$. This construction can be extended to allow for outliers by introducing a slack variable ξ_i giving rise to the following criterion:

$$\min \left[\frac{1}{2} \left|\left|\mathbf{w}\right|\right|^2 + \; C \; \sum_{i=1}^{m} \xi_i + b \right] \tag{23}$$

subject to:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq -\xi_i \tag{24}$$

with $\xi_i \geq 0$.

After kernel substitution the dual formulation involves minimisation of:

$$W(\alpha) = \frac{1}{2} \sum_{i,k=1}^{m} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{25}$$

subject to:

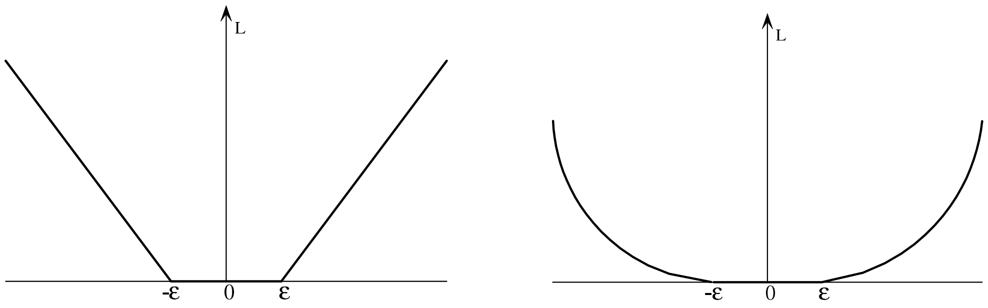


Fig. 3. Left figure: a linear ϵ-insensitive loss function versus $y_i - \mathbf{w}\cdot\mathbf{x}_i - b$. Right figure: a quadratic ϵ-insensitive loss function.

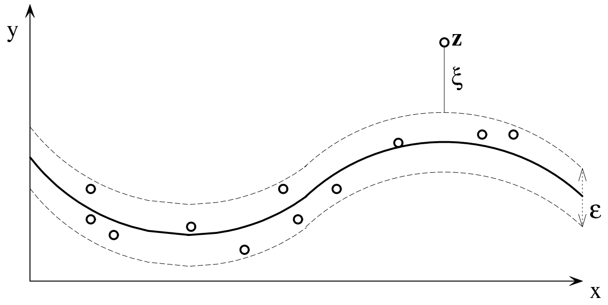


Fig. 4. The ϵ-insensitive band around a nonlinear regression function. The variables ξ measure the cost of training errors corresponding to points outside the band e.g. $\mathbf{z}$.

$$0 \le \alpha_i \le C \qquad \sum_{i=1}^{m} \alpha_i = 1 \tag{26}$$

To determine the bias we find an example, k say, which is non-bound (α_i and β_i are nonzero and $0 < \alpha_i < C$) and determine b from:

$$b = -\sum_{j=1}^{m} \alpha_j K(\mathbf{x}_j, \mathbf{x}_k) \tag{27}$$

The support of the distribution is then modeled by the decision function:

$$f(\mathbf{z}) = \mathrm{sign}\left(\sum_{j=1}^{m} \alpha_j K(\mathbf{x}_j, \mathbf{z}) + b\right) \tag{28}$$

which is again of the form (1).

2.4 Regression.

For real-valued outputs the learning task can also be theoretically motivated from statistical learning theory [17]. In this case we can use as loss functions in eq. (2) the L_2 error $(y_i - f(\mathbf{x}_i))^2$ and get the well known regularization networks [67, 21, 17]. The SVM regression method is for a particular choice of the loss function in 2, namely the linear $\epsilon-$*insensitive loss function* $\|y_i - f(\mathbf{x}_i)\|_\epsilon$ (Figure 3) [64].

Instead of constraints used for SVM classification we now use constraints $y_i - \mathbf{w}\cdot\mathbf{x}_i - b \le \epsilon$ and $\mathbf{w}\cdot\mathbf{x}_i + b - y_i \le \epsilon$ to allow for a deviation ϵ between the eventual targets y_i and the function $f(\mathbf{x}) = \mathbf{w}\cdot\mathbf{x} + b$, modeling the data.

As before we would also minimize $||\mathbf{w}||^2$ to increase flatness or penalize overcomplexity. To account for training errors we introduce slack variables $\xi_i, \widehat{\xi}_i$ for the two types of training error. The effects of this choice can be visualized as a band or tube around the hypothesis function $f(\mathbf{x})$ and any points outside this tube can be viewed as training errors (Figure 4). These slack variables are zero for points inside the tube and progressively increase for points outside the tube. This general approach is called ϵ-SV regression [63] and is the most common approach to SV regression, though not the only one [64]. For a *linear* $\epsilon-$*insensitive loss function* the task is therefore to minimize:

$$\min \left[||\mathbf{w}||^2 + C \sum_{i=1}^{m} \left(\xi_i + \widehat{\xi}_i \right) \right] \tag{29}$$

subject to

$$\begin{aligned} y_i - \mathbf{w} \cdot \mathbf{x}_i - b &\leq \epsilon + \xi_i \\ (\mathbf{w} \cdot \mathbf{x}_i + b) - y_i &\leq \epsilon + \widehat{\xi}_i \end{aligned} \tag{30}$$

where the slack variables are both positive $\xi_i, \widehat{\xi}_i \geq 0$. After kernel substitution the dual objective function is:

$$W(\alpha, \widehat{\alpha}) = \sum_{i=1}^{m} y_i(\alpha_i - \widehat{\alpha}_i) - \epsilon \sum_{i=1}^{m} (\alpha_i + \widehat{\alpha}_i) - \frac{1}{2} \sum_{i,j=1}^{m} (\alpha_i - \widehat{\alpha}_i)(\alpha_j - \widehat{\alpha}_j) K(x_i, x_j) \tag{31}$$

which is maximized subject to

$$\sum_{i=1}^{m} \widehat{\alpha}_i = \sum_{i=1}^{m} \alpha_i \tag{32}$$

and:

$$0 \leq \alpha_i \leq C \qquad 0 \leq \widehat{\alpha}_i \leq C \tag{33}$$

The decision function is then:

$$f(\mathbf{z}) = \sum_{i=1}^{m} y_i (\alpha_i - \widehat{\alpha}_i) K(\mathbf{x}_i, \mathbf{z}) + b \tag{34}$$

which is again of the form (1).

We still have to compute the bias, b, and we do so by considering the KKT conditions for regression. For a linear loss function prior to kernel substitution these are:

$$\begin{aligned} \alpha_i \left(\epsilon + \xi_i - y_i + \mathbf{w} \cdot \mathbf{x}_i + b \right) &= 0 \\ \widehat{\alpha}_i \left(\epsilon + \widehat{\xi}_i + y_i - \mathbf{w} \cdot \mathbf{x}_i - b \right) &= 0 \end{aligned} \tag{35}$$

where $\mathbf{w} = \sum_{j=1}^{m} y_j (\alpha_j - \widehat{\alpha}_j) \mathbf{x}_j$, and:

$$\begin{aligned} (C - \alpha_i)\, \xi_i &= 0 \\ (C - \widehat{\alpha}_i)\, \widehat{\xi}_i &= 0 \end{aligned} \tag{36}$$

From the latter conditions we see that only when $\alpha_i = C$ or $\widehat{\alpha}_i = C$ are the slack variables non-zero: these examples correspond to points outside the ϵ-insensitive tube. Hence we can find the bias from a non-bound example with $0 < \alpha_i < C$ using $b = y_i - \mathbf{w} \cdot \mathbf{x}_i - \epsilon$ and similarly for $0 < \widehat{\alpha}_i < C$ we can obtain it from $b = y_i - \mathbf{w} \cdot \mathbf{x}_i + \epsilon$. Though the bias can be obtained from one such example it is best to compute it using an average over all points on the margin.

2.5 Query Learning

For some real-life datasets the datapoints are initially unlabelled. Since the labels of points corresponding to *non-support* vectors are not actually required for determining an optimal separating hyperplane these points do not need to be labelled. This issue is particularly important for practical situations in which labelling data is expensive or the dataset is large and unlabelled. Since SVMs construct the hypothesis using a subset of the data containing the most informative patterns they are good candidates for *active* or *selective sampling* techniques which would predominantly request the labels for those patterns which will become support vectors

During the process of active selection the information gained from an example depends both on the position (available information) and on its label (unavailable information before querying). Thus we must follow a heuristic strategy to maximize information gain at each step. Firstly we note that querying a point within the margin band (Figure 1) *always* guarantees a gain whatever the label of the point: we do not gain by querying a point outside the band unless the current hypothesis predicts the label incorrectly. In a sense the best points to query are those points which are closest to the current hyperplane [10]. Intuitively this makes sense since these are most likely to be maximally ambiguous with respect to the current hypothesis and hence the best candidates for ensuring that the information received is maximized. Hence a good strategy [10] is to start by requesting the labels for a small initial set of data and then successively querying the labels of points closest to the current hyperplane. For noiseless datasets plateauing of the dual objective function provides a good stopping criterion (since learning non-support vectors would not change the value of $W(\alpha)$), whereas for noisy datasets emptying of the margin band and a validation phase can provide a stopping criterion [10]. Finding a good stopping criterion is an open question.

2.6 Algorithmic Approaches to Training SVMs

For classification, regression or novelty detection we see that the learning task involves optimization of a quadratic cost function and thus techniques from quadratic programming are most applicable including quasi-Newton, conjugate gradient and primal-dual interior point methods. Certain QP packages are readily applicable such as MINOS and LOQO. These methods can be used to train an SVM rapidly but they have the disadvantage that the kernel matrix is stored in memory. For small datasets this is practical and QP routines are the best choice, but for larger datasets alternative techniques have to be used. These split into two categories: techniques in which kernel components are evaluated and discarded during learning and *working set* methods in which an evolving subset of data is used. For the first category the most obvious approach is to sequentially update the α_i and this is the approach used by the Kernel Adatron (KA) algorithm [18]. For binary classification (with no soft margin or bias) this is a simple gradient ascent procedure on the SVM dual cost function in which $\alpha_i > 0$ initially and the α_i are subsequently sequentially updated using:

$$\alpha_i \leftarrow \beta_i \theta\left(\beta_i\right) \qquad \text{where} \qquad \beta_i = \alpha_i + \eta\left(1 - y_i \sum_{j=1}^{m} \alpha_j y_j K(\mathbf{x}_i, \mathbf{x}_j)\right) \tag{37}$$

and $\theta(\beta)$ is the Heaviside step function. The optimal learning rate η can be readily evaluated: $\eta = 1/K(\mathbf{x}_i, \mathbf{x}_i)$ and a sufficient condition for convergence is $0 < \eta K(\mathbf{x}_i, \mathbf{x}_i) < 2$. With the given SVM decision function this method is very easy to implement and can give a quick impression of the performance of SVMs on classification tasks. It is equivalent to Hildreth's method in Optimization theory and can be generalized to the case of soft margins and inclusion of a bias [32]. However, it is not as fast as most QP routines, especially on small datasets.

Rather than sequentially updating the α_i the alternative is to update the α_i in parallel but using only a subset or *chunk* of data at each stage. Thus a QP routine is used to optimize the objective function on an initial arbitrary subset of data. The support vectors found are retained and all other datapoints (with $\alpha_i = 0$) discarded. A new working set of data is then derived from these support vectors and additional datapoints which maximally violate the storage constraints. This *chunking* process is then iterated until the margin is maximized. Of course, this procedure may still fail because the dataset is too large or the hypothesis modeling the data is not sparse (most of the α_i are non-zero, say). In this case *decomposition* [40] methods provide a better approach: these algorithms only use a fixed size subset of data with the α_i for the remainder kept fixed.

The limiting case of decomposition is the Sequential Minimal Optimization (SMO) algorithm of Platt [45] in which only two α_i are optimized at each iteration. The smallest set of parameters which can be optimized with each iteration is plainly two if the constraint $\sum_{i=1}^{m} \alpha_i y_i = 0$ is to hold. Remarkably, if only two parameters are optimized and the rest kept fixed then it is possible to derive an analytical solution which can be executed using few numerical operations. The algorithm therefore selects two Lagrange multipliers to optimism at every step and separate heuristics are used to find the two members of the pair.

The SMO algorithm has been refined to improve speed [30] and generalised to cover the above three tasks of classification [45], regression [55] and novelty detection [53].

3 Applications of Support Vector Machines to machine vision.

3.1 Learning to Recognize 3-D Objects

We show the potential of SVMs in robotics addressing the recognition of 3-D objects from video images. We describe an aspect-based recognition approach using SVMs. Aspect-based recognition techniques have received increasing attention from both the psychophysical [57, 15] and computer vision [47, 1, 61, 8, 44, 37, 68, 40, 48] communities. These techniques are well-suited for recognition problems in which geometric models of the objects are difficult to obtain. This is typically the case when a robot has to interact with complex objects.

In the following, we give a brief overview of our recognition method. For more information see [48].

Aspect-Based Recognition System We build a system which is able to recognize an object in an image among a given set of objects. The system consists of five stages:

- **Data Collection:**
  The first step is to collect images of the objects we want to recognize. These images should cover a wide range of viewing angles of each object.
- **Preprocessing:**
  Each image of the objects is represented by a feature vector of fixed length. A typical representation is the pixel representation. Other representations like wavelets and principal components can be used. We discuss some of them in the next subsection.
- **Building the Training Sets:**
  The set of preprocessed images of each object is stored as a training set.
- **Training the System:**
  The SVM associated to each pair of objects is computed. If g is the number of objects, this requires to train the SVM algorithm discussed in section 2.2 on all possible pairs of objects i and j, with $i, j = 1, \ldots, g$, $i \neq j$. The number of pairs is $\frac{g(g-1)}{2}$. We denote the SVM associated to objects i and j with $f^{ij}(\mathbf{x})$.
- **Recognition:**
  Recognition is performed following the rules of a tennis tournament. Each object is regarded as a *player*, and the outcome of a *match* is determined by the SVM classifier which was trained to distinguish between the two objects. If the players are objects i and j, the system determines the winner according to the sign of $f^{ij}(\mathbf{x})$. For simplicity, we assume that there are 2^K objects, in the first round 2^{K-1} matches are played and the 2^{K-1} losing players are eliminated. The 2^{K-1} winners advance to the second round. The $(K-1)$-th round is the final between the remaining 2 players that won all the previous matches. Overall, this procedure requires $2^K - 1$ classifications. As mentioned in section 2.2, other multiclass classification approaches can be used for such problems.

In [48] we used the COIL (Columbia Object Image Library) database[1], a standard databases for 3-D object recognition. It consists of 7200 color images of 100 objects (72 views for each of the 100 objects). As explained in detail in [37], the objects are positioned in the center of a turntable and observed from a fixed viewpoint. For each object, the turntable is rotated 72 times in steps of $5°$.

Figure 5 shows a selection of the objects in the database. Figure 6 shows different poses of a specific object, one every $45°$.

In [48], each initial color image was first transformed into a 32×32 gray-level image leading to a feature vector of $32 \times 32 = 1024$ components. The gray values were in the range between 0 and 255.

Each training set contained 36 views of the same object. The remaining 36 images per object were used to test the system. For each pair of objects i, j a linear SVM $f^{ij}(\mathbf{x}) = \mathbf{w}^{ij} \cdot \mathbf{x} + b^{ij}$ was computed.

The number of support vectors was ranging between 30% and 60% of the initial 72 training images for each object pair. This large percentage of support vectors was due to the high dimensionality of the feature space in combination with the small number of examples.

The system recoginzed most of the 32 objects with 100% recognition rate. The system maintained its performance after adding substantial amount of random additive noise to the test images (see [48] for details).

[1] Available via anonymous ftp at `www.cs.columbia.edu`.

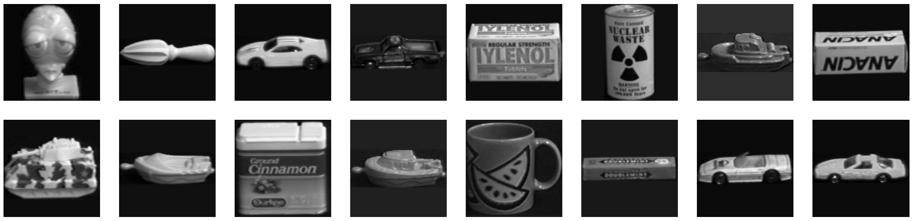


Fig. 5. Images of 16 of the COIL objects.

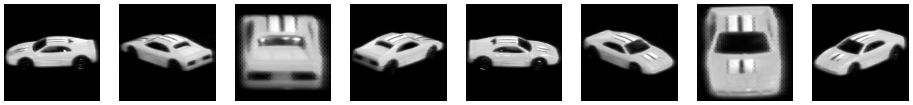


Fig. 6. Eight of the 72 images of a COIL object.

3.2 Learning to detect objects

Detection of real-world objects in images, such as faces and people, is a problem of fundamental importance in many areas of image processing and robotics: for autonomous navigation, obstacles and landmarks need to be detected; for face recognition, for example for human-computer/robot interaction, the face must first be detected before being recognized; effective indexing into image and video databases/memory relies on the detection of different classes of objects. The detection of objects poses challenging problems: the objects are difficult to model, there is significant variety in color and texture, and 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 handcrafted parameterized curves, [5, 71]. 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 [60, 31, 25, 49, 70, 23, 34]. 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[56] and Vaillant, et al. [62] for the detection of frontal faces in cluttered scenes, with similar architectures later used by Moghaddam and Pentland [36], Rowley, et al. [51], and Osuna et al. [40]. We now briefly discuss how the learning mechanisms outlined in the first part of the paper can be used as an approach to object detection in images.

A trainable system for object detection We briefly describe a trainable system for object detection. For more information on the system we refer the reader to [16]. The system is based on [43] and can be used to learn any class of objects. The overall framework has been motivated and successfully applied in the past [43]. The system consists of three parts:

- A set of (positive) example images of the object class considered (i.e. images of frontal faces) and a set of negative examples (i.e. any non-face image) are collected.
- The images are transformed into vectors in a chosen representation (i.e. a vector of the size of the image with the values at each pixel location).
- The vectors (examples) are used to train a SVM classifier to learn the classification task of separating positive from negative examples. A new set of examples is used to test the system. The full architecture involves scanning an (test) image over different positions and scales.

Two choices need to be made: the representation in the second stage, and the kernel of the SVM (discussed above) in the third stage. In [16] various image representations were evaluated:

- The *pixel representation*: train an SVM using the raw pixel values of the images (possibly scaled between 0 and 1).
- The *eigenvector (principal components) representation:* compute the correlation matrix of the positive examples (the pixel vectors corresponding to images of the positive class - say images of faces) 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 eigenvectors, or use the projections on only the first few principal components.

– The *wavelet representation:* consider a set of Haar wavelets at different scales and locations, and compute the projections of the images on the chosen wavelets. Wavelets at different scales can be used.

Once a representation is chosen and an SVM is trained, the system can be used for detecting objects in new images (i.e. in the environment where a robot navigates) by simply scanning the images with a window of size equal to that of the training data (possibly scaling the images at various scales), passing the scanned window of the image to the trained SVM and deciding at each location in the image whether or not there is an object of the positive class. For many classes of objects, mutliclass SVMs can be used in the same manner.

An important issue is that of feature selection. For example, in the case that the pixel values of the images are used to train a classifier, the question is which pixels (that is, which parts of the images) are more important? Finding good methods for feature selection is a difficult problem. A heuristic is suggested in [16], and more formal methods are suggested in [26]. Selecting features can be important when many features are available, like in the case of image, speech, or video processing. In fact many learning methods suffer the ”curse of dimensionality”. It turns out that an important characteristic of SVM and kernel machines developed within the statistical learning theory framework is that they can handle large dimensional data. This is well explained within the theory [64], and also experimentally verified by many researchers (see for example [16]). This characteristic of SVMs makes them suitable learning methods for complex tasks where many features are necessary, which is often the case for robotics applications.

4 Conclusion

In this paper we outlined an approach to solving complex problems based on kernel learning methods. These methods are motivated and justified theoretically within the well established field of statistical learning theory [64], and can be used for a number of learning tasks, such as classification, regression, and novelty detection. The methods are shown to work well in practice and, unlike other methods such as neural networks, they lead to learning machines that can be trained efficiently, with few parameters and unique optimal solutions.

Complex systems, such as autonomous robots, need to handle all or some of the learning tasks outlined in this paper. We beleive that combining the methods discussed here in order to build such systems is a promising direction for research.

References

1. S. Akamatsu, T. Sasaki, H. Fukumachi, and Y. Suenaga, “A robust face identification scheme - KL expansion of an invariant feature space,” *SPIE Proc. Vol. 1607: Intelligent Robots and Computer Vision X: Algorithms and Techniques*, 1991, pp. 71–84.
2. M. Anthony and P. Barlett. *Learning in neural networks: theoretical foundations.* Cambridge University Press, 1999.
3. N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 686:337–404, 1950.
4. N. Barabino, M. Pallavicini, A Petrolini, M. Pontil and A. Verri. Support vector machines vs multi-layer perceptrons in particle identification. In *Proceedings of the European Symposium on Artifical Neural Networks ’99* (D-Facto Press, Belgium), p. 257-262, 1999.
5. 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.
6. V. Blanz, B. Scholkopf, H. Bulthoff, C. Burges, V.N. Vapnik, and T. Vetter, “Comparison of view–based object recognition algorithms using realistic 3D models,” *Proc of ICANN’96*, LNCS Vol. 1112, 1996, pp. 251–256.
7. M. Brown, W. Grundy, D. Lin, N. Cristianini, C. Sugnet, M. Ares Jr. and D. Haussler. Support vector machine classification of microarray gene expression data. University of California, Santa Cruz, Technical Report UCSC-CRL-99-09.
8. R. Brunelli and T. Poggio, “Face Recognition: Features versus Templates,” *IEEE Trans. on PAMI*, Vol. 15, 1993, pp. 1042–1052.
9. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2, p. 121-167, 1998.
10. C. Campbell, N. Cristianini and A. Smola. Query Learning with Large Margin Classifiers. *Proceedings of the Seventeenth International Conference on Machine Learning*, ed. Pat Langley, Morgan Kaufmann, 2000, p. 111-118..
11. O. Chapelle and V. Vapnik. Model selection for support vector machines, to appear in *Advances in Neural Information Processing Systems* 12, ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press, 2000.
12. C. Cortes and V. Vapnik. Support vector networks. *Machine Learning* 20, p. 273-297, 1995.
13. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other Kernel-based Learning Methods.* Cambridge University Press, to appear January 2000.
14. H. Drucker, C. Burges, L. Kaufman, A. Smola and V. Vapnik. Support vector regression machines. In: M. Mozer, M. Jordan, and T. Petsche (eds.). *Advances in Neural Information Processing Systems*, 9, MIT Press, Cambridge, MA, 1997.

15. S. Edelman, H. Bulthoff and D. Weinshall, “Stimulus Familiarity Determines Recognition Strategy for Novel 3-D Objects,” AI Memo No. 1138, MIT, Cambridge, 1989.
16. T. Evgeniou, C. Papageorgiou, M. Pontil, and T. Poggio. Image representations for object detection using kernel classifiers. In *Proceedings ACCV*, Taiwan, January 2000.
17. T. Evgeniou, M. Pontil, and T. Poggio. Regularization Networks and Support Vector Machines. Advances in Computational Mathematics 13 (2000) 1, pages 1-50.
18. T.-T. Friess, N. Cristianini and C. Campbell. The kernel adatron algorithm: a fast and simple learning procedure for support vector machines. *15th Intl. Conf. Machine Learning*, Morgan Kaufman Publishers, p. 188-196, 1998.
19. A. Gersho and R.M. Gray. *Vector quantization and signal compression.* Kluwer Academic Publishers, Boston, 1991.
20. 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.
21. F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7:219–269, 1995.
22. I. Guyon, N. Matic and V. Vapnik. Discovering informative patterns and data cleaning. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, MIT Press, p. 181–203, 1996.
23. B. Heisele, U. Kressel, and W. Ritter. Tracking Non-rigid, Moving Objects Based on Color Cluster Flow. In *Computer Vision and Pattern Recognition*, 1997.
24. R. Herbrich, T. Graepel and C. Campbell. Bayesian learning in reproducing kernel Hilbert spaces, submitted to *Machine Learning*, 1999.
25. D. Hogg. Model-based vision: a program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.
26. T. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proc. of Neural Information Processing Conference*, 1998.
27. A. K. Jain. *Fundamentals of digital image processing.* Prentice-Hall Information and System Sciences Series, New Jersey, 1989.
28. T. Joachims. Estimating the Generalisation Performance of an SVM Efficiently. *Proceedings of the Seventeenth International Conference on Machine Learning*, ed. Pat Langley, Morgan Kaufmann, 2000, p. 431-438.
29. T. Joachims. Text Categorization with Support Vector Machines. Technical Report LS-8 Report 23, University of Dortmund, November 1997.
30. S. Keerthi, S. Shevade, C. Bhattacharyya and K. Murthy. Improvements to Platt’s SMO algorithm for SVM classifier design. Tech Report, Dept. of CSA, Banglore, India, 1999.
31. M.K. Leung and Y-H. Yang. A region based approach for human body analysis. *Pattern Recognition*, 20(3):321–39, 1987.
32. D. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, 1984.
33. E. Mayoraz and E. Alpaydin. Support Vector Machines for Multiclass Classification, *Proceedings of the International Workshop on Artifical Neural Networks (IWANN99)*, IDIAP Technical Report 98-06, 1999.
34. 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.
35. S. Mika, G. Ratsch, J. Weston, B. Schölkopf and K.-R. Muller. Fisher Discriminant Analysis with Kernels. *Proceedings of IEEE Neural Networks for Signal Processing Workshop* 1999, 8 pages, 1999.
36. B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. Technical Report 326, MIT Media Laboratory, 1995.
37. H. Murase and S.K. Nayar, “Visual Learning and Recognition of 3-D Object from Appearance,” *Int. J. Comput. Vision*, Vol. 14, 1995, pp. 5–24.
38. 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.
39. ORL dataset: Olivetti Research Laboratory, 1994, http://www.orl.co.uk/facedatabase.html.
40. E. Osuna, R. Freund, and F. Girosi. Support Vector Machines: Training and Applications. A.I. Memo 1602, MIT Artificial Intelligence Laboratory, 1997.
41. E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *Computer Vision and Pattern Recognition*, pages 130–36, 1997.
42. C. Papageorgiou, F. Girosi, and T.Poggio. Sparse correlation kernel based signal reconstruction. A.I. Memo 1635, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1998. (CBCL Memo 162).
43. C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Proceedings of the International Conference on Computer Vision*, Bombay, India, January 1998.
44. A. Pentland, B. Moghaddam, and T. Starner, “View-based and modular eigenspaces for face recognition,” *Proc. CVPR*, 1994, pp. 84–91.
45. J. Platt. Fast training of SVMs using sequential minimal optimisation. In B. Schölkopf, C.Burges and A. Smola (ed.), *Advances in Kernel Methods: Support Vector Learning*, MIT press, Cambridge, MA, p. 185-208, 1999.
46. J.Platt, N. Cristianini and J. Shawe-Taylor. Large Margin DAGS for Multiclass Classification. In *Advances in Neural Information Processing Systems*, 12 ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press, 2000.

47. T. Poggio and S. Edelman, “A Network that Learns to Recognize Three-Dimensional Objects,” *Nature*, Vol. 343, 1990, pp. 263-266.
48. M. Pontil and A. Verri, “Support Vector Machines for 3D Object Recognition,” *IEEE Trans. on PAMI*, Vol. 20, No. 6, 1998, pp. 637-646.
49. K. Rohr. Incremental recognition of pedestrians from image sequences. *Computer Vision and Pattern Recognition*, pages 8–13, 1993.
50. D. Roobaert. Improving the Generalisation of Linear Support Vector Machines: an Application to 3D Object Recognition with Cluttered Background. *Proc. Workshop on Support Vector Machines at the 16th International Joint Conference on Artificial Intelligence*, July 31-August 6, Stockholm, Sweden, p. 29-33 1999.
51. H.A. Rowley, S. Baluja, and T. Kanade. Human face detection in visual scenes. Technical Report CMU-CS-95-158, School of Computer Science, Carnegie Mellon University, July/November 1995.
52. B. Schölkopf, C. Burges and A. Smola. *Advances in Kernel Methods: Support Vector Machines.* MIT Press, Cambridge, MA. 1998.
53. B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson. Estimating the support of a high-dimensional distribution. Microsoft Research Corporation Technical Report MSR-TR-99-87, 1999.
54. J. Shawe-Taylor and N. Cristianini. Margin distribution and soft margin. In A. Smola, P. Barlett, B. Schölkopf and C. Schuurmans (eds), *Advances in Large Margin Classifiers*, Chapter 2, MIT Press, 1999.
55. A. Smola and B. Schölkopf. A tutorial on support vector regression. NeuroColt2 TR 1998-03, 1998.
56. K-K. Sung and T. Poggio. Example-based learning for view-based human face detection. A.I. Memo 1521, MIT Artificial Intelligence Laboratory, December 1994.
57. M. Tarr and S. Pinker, “Mental Rotation and Orientation-Dependence in Shape Recognition,” *Cognitive Psychology*, Vol. 21, 1989, pp. 233-282.
58. D. Tax and R. Duin. Data domain description by Support Vectors. In *Proceedings of ESANN99*, ed. M Verleysen, D. Facto Press, Brussels, p. 251-256, 1999.
59. M. Tipping. The Relevance Vector Machine. In *Advances in Neural Information Processing Systems, 12*. S. Solla, T. Leen nd K.-R. Muller (eds) MIT Press, Cambridge, MA, 2000 (to appear).
60. 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.
61. M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proceedings CVPR*, pages 586–591, Hawaii, June 1991.
62. R. Vaillant, C. Monrocq, and Y. Le Cun. Original approach for the localisation of objects in images. *IEE Proceedings Vision Image Signal Processing*, 141(4):245–50, August 1994.
63. V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, N.Y., 1995.
64. V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.
65. 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.
66. M. Vidyasagar. *A theory of learning and generalisation.* Springer-Verlag, Berlin, 1997.
67. G. Wahba. *Spline Models for Observational Data.* Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.
68. J. Weng, “Cresceptron and SHOSLIF: Toward comprehensive visual learning.” In S.K. Nayar and T. Poggio eds *Early Visual Learning* (Oxford Univ. Press, 1996).
69. J. Weston and C. Watkins. Multi-Class Support Vector Machines, in *Proceedings of ESANN99*, ed. M Verleysen, D. Facto Press, Brussels, p. 219-224, 1999.
70. C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. Technical Report 353, MIT Media Laboratory, 1995.
71. A. Yuille, P. Hallinan, and D. Cohen. Feature Extraction from Faces using Deformable Templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.