Machine Learning, Machine Vision, and the Brain

Tomaso Poggio and Christian R. Shelton

Center for Biological and Computational Learning
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139, USA

September 27, 1999

Abstract

The problem of learning is arguably at the very core of the problem of intelligence, both biological and artificial. In this paper we review our work over the last ten years in the area of supervised learning, focusing on three interlinked directions of research: theory, engineering applications (making intelligent software) and neuroscience (understanding the brain's mechanisms of learning) which contribute to and complement each other.

Keywords: Machine Learning, Regularization, Support Vector Machine,
IT Cortex, Function Approximation, Object Detection

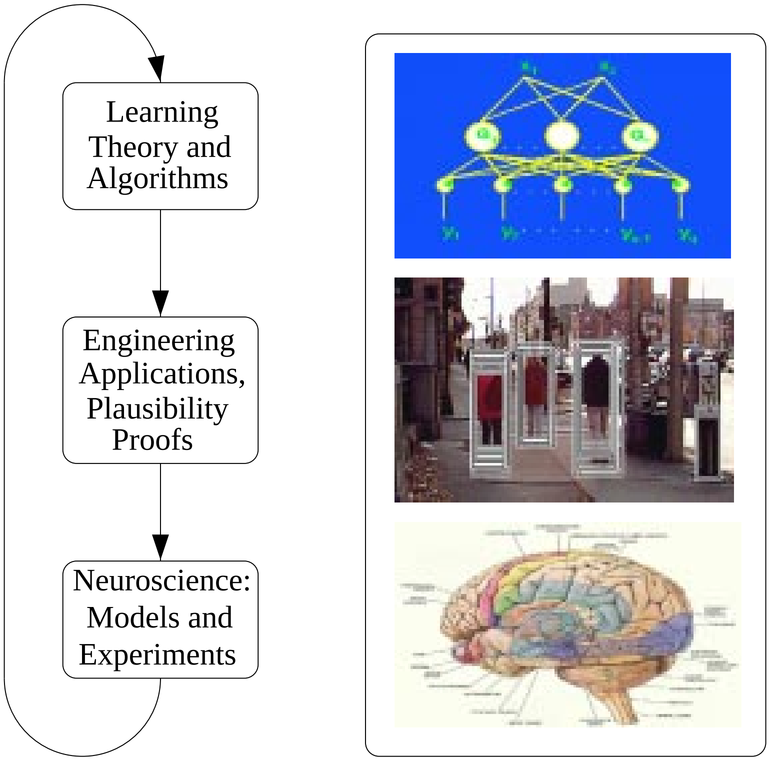
Learning Theory and Algorithms
Engineering Applications, Plausibility Proofs
Neuroscience: Models and Experiments

Figure 1: A multidisciplinary approach to supervised learning

1 Introduction

Learning is now perceived as a gateway to understanding the problem of intelligence. Since seeing is intelligence, learning is also becoming a key to the study of artificial and biological vision. In the last few years both computer vision – which attempts to build machines that see – and visual neuroscience – which aims to understand how our visual system works – are undergoing a fundamental change in their approaches. Visual neuroscience is beginning to focus on the mechanisms which allow the cortex to adapt its circuitry and learn a new task. Instead of building a hardwired machine or program to solve a specific visual task, computer vision is trying to develop systems that can be trained with examples of any of a number of visual tasks. Vision systems that *learn and adapt* represent one of the most important directions in computer vision research. This reflects an overall trend – to make intelligent systems that do not need to be fully and painfully programmed. It may be the only way to develop vision systems that are robust and easy to use in many different tasks.

Building systems without explicit programming is not a new idea. Ex-

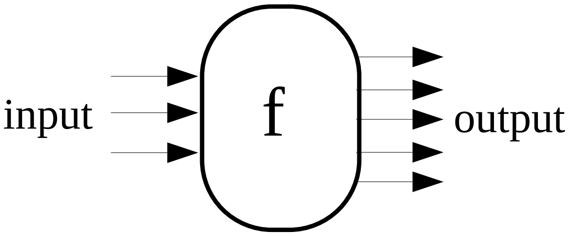

Figure 2: In the learning-from-examples paradigm, we learn a function f from input-output pairs $(\mathbf{x}_i, \mathbf{y}_i)$ called the training set.

tensions of the classical pattern recognition techniques have provided a new metaphor – learning from examples – that makes statistical techniques more attractive (for an overview of machine learning and other applications, see Mitchell (1997)). As a consequence of this new interest in learning, we are witnessing a renaissance of statistics and function approximation techniques and their applications to domains such as computer vision. In this paper we review our work over the last ten years in the area of supervised learning, focusing on three interlinked directions of research sketched in figure 1: theory, engineering applications (making intelligent software), and neuroscience (understanding the brain's mechanisms of learning). The figure shows an ideal continuous loop from theory to feasibility demonstrations to biological models feeding back into new theoretical ideas. In reality, the interactions – as one may expect – are less predictable but not less useful. For instance in 1990, ideas from the mathematics of learning theory – Radial Basis Function Networks – suggested a model for biological object recognition which led to the physiological experiments incortex described later in the paper. It was only later that the same idea found its way into the computer graphics applications described in the conclusions.

2 Learning and Regularization

In this article we will concentrate on one aspect of learning: *supervised learning*. Supervised learning – or *learning-from-examples* – refers to systems that are trained, instead of programmed, by a set of examples, that is input-output pairs $(\mathbf{x}_i, \mathbf{y}_i)$ as sketched in figure 2. At run-time they will hopefully provide a correct output for a new input not contained in the training set. One way to set the problem of learning-from-examples in a mathematically well-founded framework is the following. Supervised learning can be regarded as the regression problem of interpolating or approximating a multivariate function from sparse data (figure 3). The data are the examples. Generalization means estimating the value of the function for points in the input space in which data are not available.

Once the ill-posed problem of learning-from-examples has been formulated

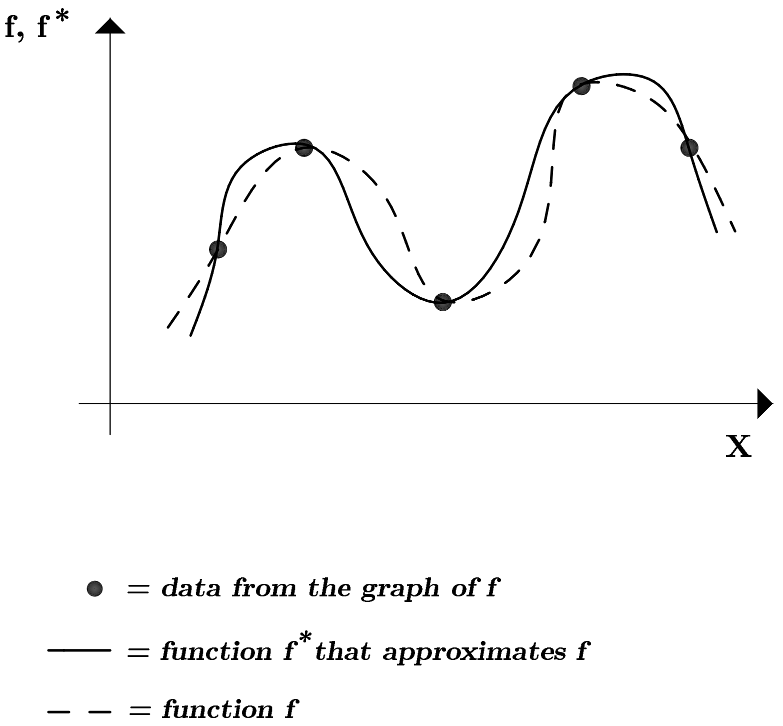

Figure 3: *Learning-from-examples as multivariate function approximation or interpolation from sparse data. Generalization means estimating* $f^*(x) \approx f(x)$ *,*$\forall x \in X$ *from the examples* $f^*(x_i) = f(x_i)$*,* $i = 1, \ldots, N$*.*

as a problem of function approximation, an obvious approach to solving it is *regularization*. Regularization solves the problem of choosing among the infinite number of functions that all pass through the finite number of data points by imposing a smoothness constraint on the final solution (as we describe below, it is reasonable to assume that any learnable function is smooth). This results in minimizing the cost functional

$$H[f] = \sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 + \lambda \|f\|_K^2 \tag{1}$$

where $\|f\|_K^2$ is a measure of deviation from smoothness of the solution f (see Wahba (1990) and Evgeniou, Pontil, and Poggio (1999)) and the sum is the deviation of the function from the data points (thus we are making a tradeoff between accurately modeling the data points and the smoothness of the learned function). For instance in the one-dimensional case, using $\|f\|_K^2 = \int dx \left(\frac{\partial^2 f(x)}{\partial x^2}\right)^2$ in H yields cubic splines as the minimizer $f(x)$ of H.

The use of smoothness stabilizers in the functional equation 1 penalizing non-smooth functions can be justified by observing that it would be impossible to generalize for input-output relations that are not smooth, that is for cases in which "similar" inputs do not correspond to "similar" outputs (in an appropriate metric!). Such cases exist: for instance the mapping provided by a telephone directory between names and telephone numbers is usually not "smooth" and it is a safe bet that it would be difficult to learn it from examples!

The functional regularization approach can also be regarded from a probabilistic and Bayesian perspective. In particular, as Girosi, Jones, and Poggio (1995) and Girosi, Jones, and Poggio (1993) (see also Poggio and Girosi (1990b), Poggio and Girosi (1990a), and Wahba (1990)) describe, an empirical Bayes approach leads to the maximum *a posteriori* (MAP) estimate of

$$P(f|g) \propto P(f)\ P(g|f)\ ,$$

where the set $g = (\mathbf{x}_i, \mathbf{y}_i)_{i=1}^{N}$ consists of the input-output pairs of training examples and f is again the learned function. Under a few assumptions (additive Gaussian noise and a linear Gaussian prior), taking this probabilistic approach to solving the learning problem is equivalent to minimizing equation 1.

2.1 Regularization is equivalent to feed forward networks: Regularization Networks

A key result for our work since 1990 is that, under rather general conditions, the solution of the regularization formulation of the approximation problem can be expressed as the linear combination of basis functions, centered on the data points and depending on the input $\mathbf{x}$. The form of the basis function K depends on the specific smoothness criterion, that is the functional $|f|_K^2$. The simplest solution (for several important K such as the Gaussian) is

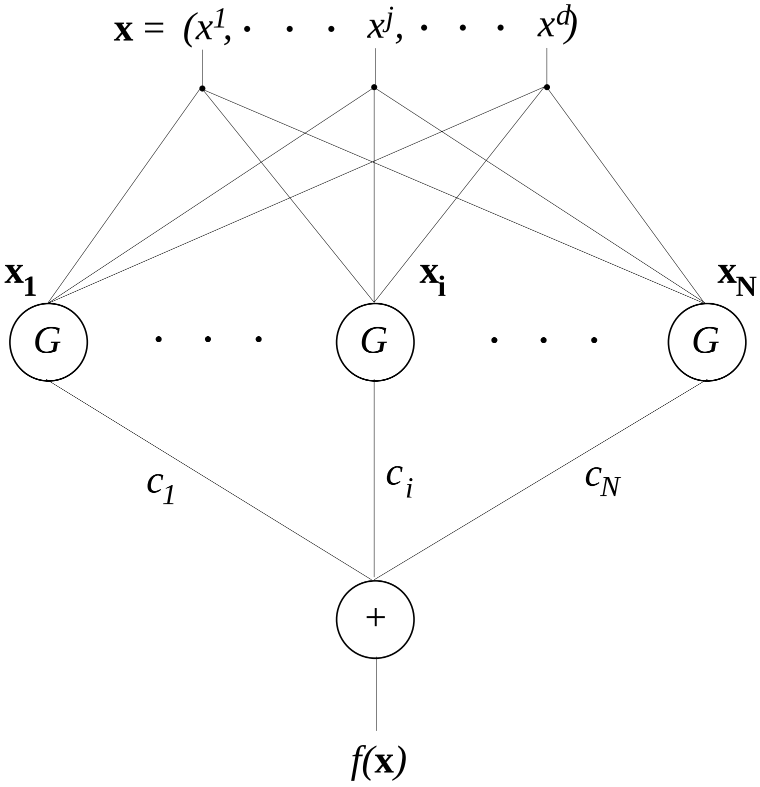

Figure 4: *A Regularization Network. The input vector* $\mathbf{x}$ *is d-dimensional, there are N hidden units, one for each example* $\mathbf{x}_i$*, and the output is a scalar function* $f(\mathbf{x})$*.*

$$f(\mathbf{x}) = \sum_{i=1}^{l} c_i K(\mathbf{x}, \mathbf{x}_i) \qquad (2)$$

As observed by Poggio and Girosi (1990b) (see also Broomhead and Lowe (1988)), the solution provided by equation 2 can always be rewritten as a network with one hidden layer containing as many units as examples in the training set (see figure 4). We called these networks Regularization Networks (RN). The coefficients c_i that represent the "weights" of the connections to the output are "learned" by minimizing the functional H over the training set (Girosi, Jones, and Poggio 1995).

2.2 Radial Basis Functions

An interesting special case arises for radial K. Radial Basis Function techniques – or Radial Basis Function networks (RBFs) (Powell 1987; Micchelli 1986; Poggio and Girosi 1989; Girosi, Jones, and Poggio 1995) follow from regularization when $K(\mathbf{s}, \mathbf{t})$ is shift invariant and radially symmetric: the best example is a Gaussian $K(\mathbf{s}, \mathbf{t}) = G_\sigma(|\mathbf{s} - \mathbf{t}|^2)$:

$$f(\mathbf{x}) = \sum_{i=1}^{l} c_i G_\sigma(|\mathbf{x} - \mathbf{x}_i|^2). \tag{3}$$

In the Gaussian case, these RBF networks consist of units each tuned to one of the examples with a bell-shaped activation curve. In the limit of very small σ for the variance of the Gaussian basis functions, RBF networks become look-up tables. Thus

- Each "unit" computes the distance $\|\mathbf{x} - \mathbf{x}_i\|$ of the input vector $\mathbf{x}$ from its center $\mathbf{x}_i$ and
- in the limiting case of G being a very narrow Gaussian, the network becomes a *look-up* table
- centers are like *templates*

Gaussian RBF networks are a simple extension of look-up tables and can be regarded as interpolating look-up tables, providing a very simple interpretation to the result of relatively sophisticated mathematics . The "vanilla" RBF described above can be generalized to the case in which there are fewer units than data and the centers $\mathbf{x}_i$ are to be found during the learning phase of minimizing the cost over the training set. These generalized RBF networks have sometimes been called HyperBF networks (Poggio and Girosi 1990a).

3 Support Vector Machines

3.1 Regularization provides a general theory

Several representations for function approximation and regression as well as several Neural Network architectures can all be derived from regularization principles with somewhat different prior assumptions on the smoothness of the function space (that is different stabilizers, defined by different kernels K). They are therefore quite similar to each other.

Figure 5 tries to make the point that Regularization Networks provide a general framework for a number of classical and new learning techniques. In particular, the radial class of stabilizer is at the root of the techniques on the left branch of the diagram: RBF can be generalized into HyperBF and into so-called kernel methods and various types of multidimensional splines. A class of priors combining smoothness and additivity (Girosi, Jones, and Poggio 1995)

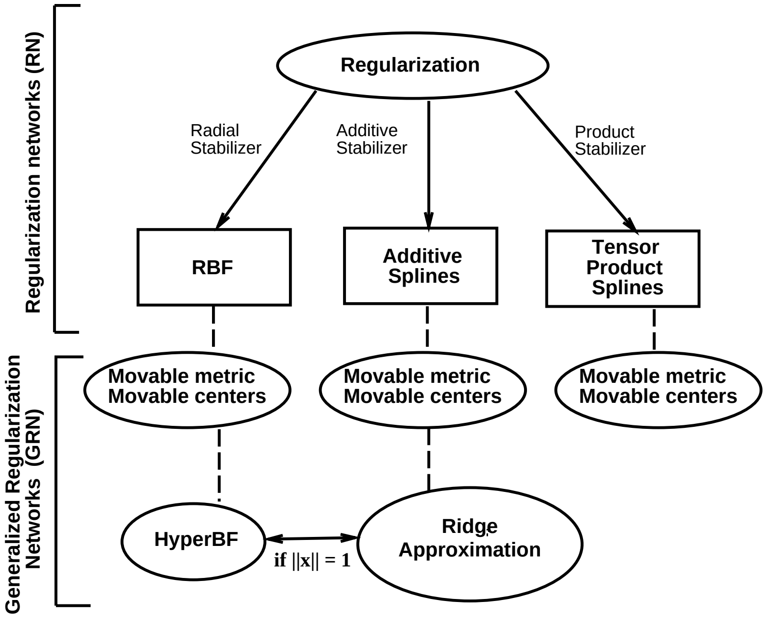
Regularization networks (RN)
Regularization
Radial Stabilizer
Additive Stabilizer
Product Stabilizer
RBF
Additive Splines
Tensor Product Splines
Generalized Regularization Networks (GRN)
Movable metric Movable centers
Movable metric Movable centers
Movable metric Movable centers
HyperBF
if ||x|| = 1
Ridge Approximation

Figure 5: Several classes of approximation schemes and corresponding network architectures can be derived from regularization with the appropriate choice of smoothness priors and associated stabilizers and basis functions, showing the common Bayesian roots. From Girosi, Jones, and Poggio (1993).

is at the root of the middle branch of the diagram: additive splines of many different forms generalize into ridge regression techniques, such as the representations used in Projection Pursuit Regression (Friedman and Stuetzle 1981), hinges (Breiman 1993), and several multilayer perceptron-like networks (with one hidden layer).

The mathematical results (Girosi, Jones, and Poggio 1995) summarized in figure 5 are useful because they provide

1. an understanding of what many different Neural Networks do and what is the function of their hidden units,
2. an approximate "equivalence" of many different schemes for regression, while giving insights into their slightly different underlying (smoothness) assumptions, and
3. a general theory for a broad class of supervised learning architectures.

3.2 Support Vector Machines and Regularization

Recently a new learning technique has emerged and become quite popular because of its good performance and its deep theoretical foundations: Support Vector Machines (SVM), proposed by Vapnik (Vapnik 1995). It is natural to ask the question of its relation with Regularization Networks. The answer is that it is very closely connected to regularization (Girosi 1998; Evgeniou, Pontil, and Poggio 1999): it can be regarded as the same type of network, corresponding to exactly the same type of solution f (that is equation 2) but "trained" in a different way and therefore with different values of the weight c_i after the training (Evgeniou, Pontil, and Poggio 1999). In particular, in SVM many of the coefficients c_i are usually zero: the $\mathbf{x}_i$ corresponding to the non-zero coefficients are called *support vectors* and capture all the relevant information of the full training set.

3.3 Support Vector Machines and Sparsity

In recent years, there has been a growing interest in using sparse function approximators. An analogy to human speech due to Stefan Mallat (of wavelet fame) provides the right intuition. If one were to describe a concept using a small dictionary of only three thousand English words, the description of most concepts would require long sentences using all of most of the three thousand words. However, if one were to describe a concept using a large dictionary of one hundred thousand words, only a small number of the words would be required for most concepts.

As we mentioned, in SVMs many of the weights c in the sum of equation 2 are zero. The link to sparsity can be made formal: Girosi (1998) proved that, loosely speaking, the sparsest representation (in a certain sense, see Girosi (1998)) is also the one with the best prediction and generalization abilities. The result suggests that a sparse representation of a signal (for instance images) from a large dictionary of features is optimal for generalization.

Finally, it is important to observe that until now the functionals of classical regularization have lacked a rigorous justification for a finite set of training data. Vapnik's seminal work has laid the foundations for a more general theory that justifies a broad range of regularization functionals for learning from finite sets, including classical regularization and Support Vector Machines for regression and for classification. The basic idea is that for a finite set of training examples the search for the best model or approximating function has to be constrained to an appropriately "small" hypothesis space (which can also be thought of as a space of machines or models or network architectures). Vapnik's theory characterizes and formalizes these concepts in terms of the *capacity* of a set of functions and *capacity control* depending on the training data: for instance, for a small training set, the capacity of the function space in which f is sought has to be small whereas it can increase with a larger training set. A key part of the theory is to define and bound the capacity of a set of functions. Evgeniou et al. (1999) show how different learning techniques based on the minimization of the H functionals listed earlier can be justified using a slight extension of the tools and results of Vapnik's statistical learning theory.

4 Object Detection with Support Vector Machines

So, one can only ask, "does all of the theory mean anything?" The mathematics of the previous section suggest that a sparse regularization network (such as a support vector machine) will perform well in classification tasks.

We present here two systems based on the theory outlined in the previous sections – they use Support Vector Machines classifiers of the form of figure 4 and equation 2 – that learn to detect and classify objects of a specific class in complex image and video sequences. In both systems, the goal is to take an image and find whether and where the object of interest is in the image.

Both use the same architecture (depicted in figure 6). A window is translated across the image. At each translation step, the sub-window of the image masked by the sliding window is fed into a feature extractor (which may return features of the image or just the raw pixel values) whose output is then given to a support vector classifier. This classifier was previously trained using labeled examples of subimages. To achieve detection at multiple scales, the image is rescaled to different sizes and the translation rerun at the new scales. Thus, the output of the classifier on a particular subimage indicates whether the object exists at that location and scale.

4.1 Face Detection

For face detection, the goal is to identify the position and scale of all of the faces in the image. The sub-window for this task was 19x19 pixels and no feature

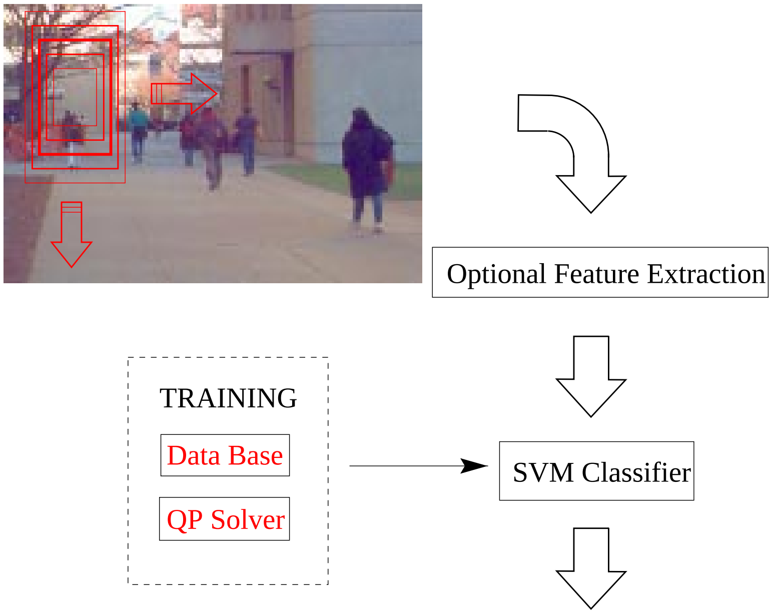

Figure 6: Architecture of SVM system for object detection

extraction was used (the gray-scale intensity values from the sub-image were fed directly to the classifier). The full system details are described in Osuna, Freund, and Girosi (1997). Here we will just quote some of the results from their experiment.

After training an SVM, most of the examples are automatically discarded because many of the c_i of equation 2 are zero. This is related to the theoretical connection between the SVM framework and sparsity and results in a network that depends only on a few "boundary" examples (the support vectors). Theoretically, these are the examples that helped to define the decision boundary. Figure 7 shows a few examples from the face detection system of Osuna *et. al.*. It is interesting to note that they appear to be the most "unfacelike" of the face images and the most "facelike" of the non-face images. Put another way, they are the most difficult training examples and the ones mostly likely to be confused later and therefore the ones which should be remembered in order to classify new examples correctly.

These learned support vectors and their associated weights were used in a network, as shown in figure 4, to do classification. Some examples of the results of the system are shown in figure 8.

4.2 Pedestrian Detection

Using the same system architecture, we can attempt to learn to detect pedestrians. Unfortunately, since pedestrians are a far more varied class of objects,

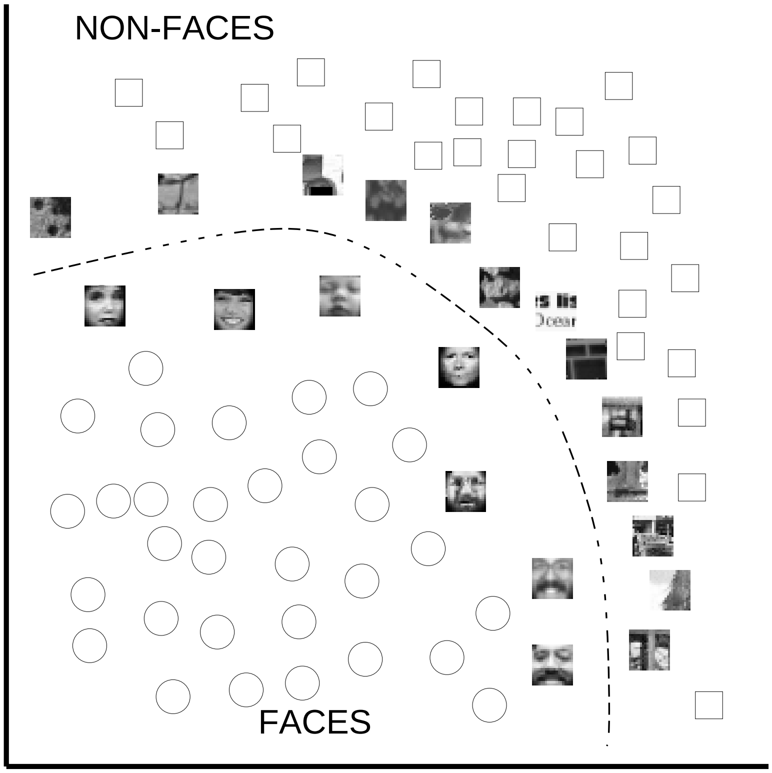

Figure 7: Some of the support vectors found by training for face detection. From Osuna, Freund, and Girosi (1997)

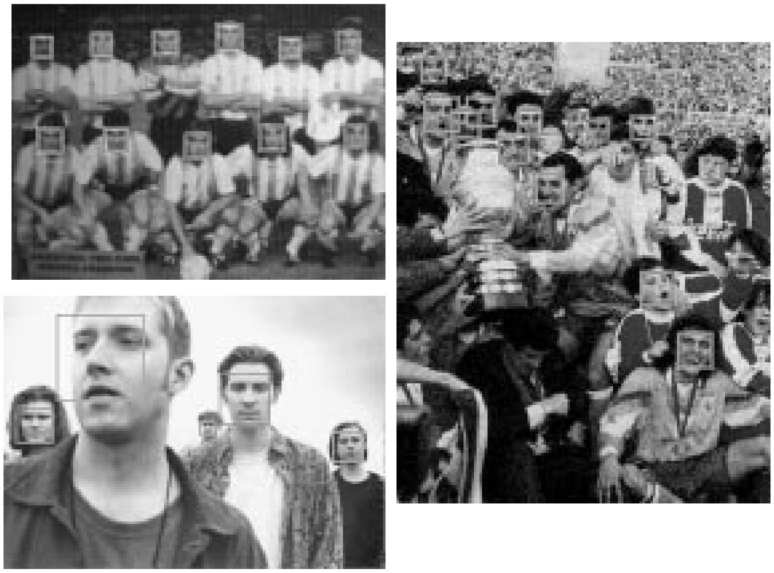

Figure 8: Results of the face-detection system of Osuna, Freund, and Girosi (1997)

using a sub-window of the pixel values is not sufficient for good performance.

To solve this problem, we add a feature extraction step (as shown in figure 6) build an overcomplete, multiscale set of the absolute values of Haar wavelets as the basic dictionary with which to describe shape. These wavelet are simple differencing filters applied to the image at different resolutions. This results in roughly 1300 coefficients for each sub-window. The full system is described in depth in Papageorgiou (1997), Oren, Papageorgiou, Sinha, Osuna, and Poggio (1997), Papageorgiou, Oren, and Poggio (1998), and Papageorgiou, Evgeniou, and Poggio (1998).

Since the "sensitivity" of the system to pedestrians can be adjusted, we can trade-off the number of undetected pedestrians (false negatives) against the number of incorrect detected non-pedestrians (false positives). Figure 9 plots a curve showing the performance of the system for various settings of the sensitivity. The upper-left corner represents an ideal system which classifies all pedestrians correctly and does not signal non-pedestrian image patches as pedestrians. These ROC curves were computed over an *out-of-sample* test set gathered around MIT and over the Internet.

The different plots in figure 9 correspond to different sets of features. Shown are the ROC curves for three systems:

- color processing with all 1326 features

- color processing with 29 features

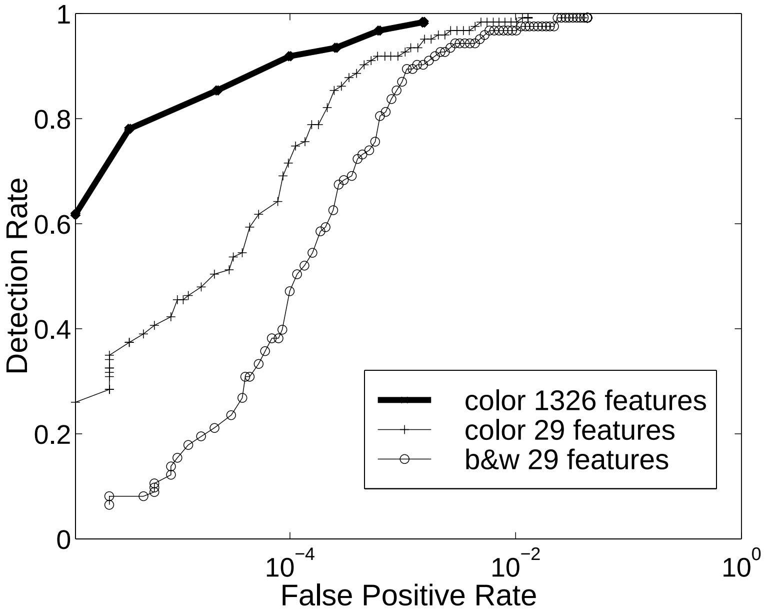

Figure 9: ROC curves for different detection systems. The detection rate is plotted against the false positive rate, measured on a logarithmic scale. The false detection rate is defined as the number of false detections per inspected window. From Papageorgiou, Evgeniou, and Poggio (1998)

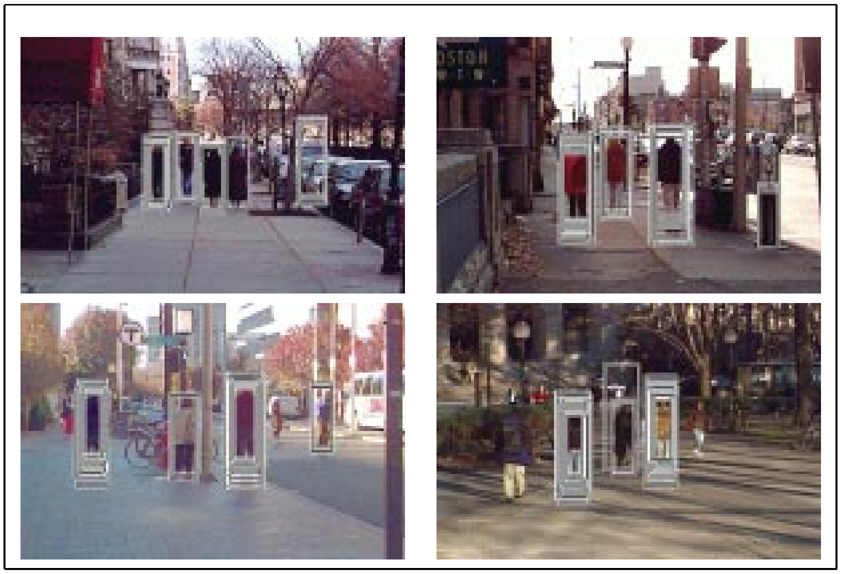

Figure 10: Results from the pedestrian detection system. Typically, missed pedestrians are due to occlusion or lack of contrast with the background. False positives can be eliminated with further training. From Papageorgiou, Evgeniou, and Poggio (1998)

- grey-level processing with 29 features

The ROC curve shows the difference in the performance resulting from the choice of features. What is not shown (for clarity) is the impact of changing the kernel function. Changes to the kernel used in the SVM had little effect on the final performance (those shown are for polynomials of degree 3). As expected, using color features results in a more powerful system. The curve of the system with *no feature selection* is clearly superior to all the others. This indicates that for the best accuracy, using all the features is optimal. When classifying using this full set of features, we pay for the accuracy through a slower system. It may be possible to achieve the same performance as the 1326 feature system with fewer features; this is an open question, however. Reducing the number of features is important to reducing the running time of the final detection system. Examples of processed images are shown in Figure 10; these images were not part of the training set.

The system has also been extended to allow detection of frontal, rear, and side views of pedestrians. It is currently installed in an experiemenal car at Daimler. Figure 11 shows the results of processing a video sequence from this car driving in downtown Ulm, Germany. The results shown here are without using any motion or tracking information; adding this information to the system would improve results. From the sequence, we can see that the system generalizes

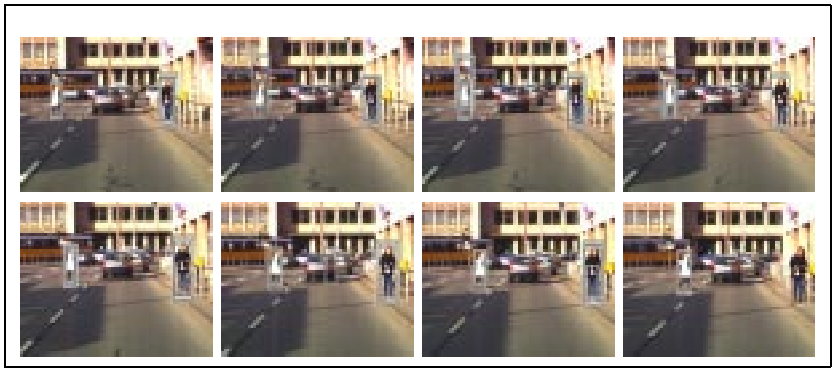

Figure 11: Processing the “Downtown Ulm” sequence with the frontal, rear, and side view detection system. The system performs the detection frame-by-frame: it uses no motion or tracking. Adding motion information and the capability of integrating detection over time improves results. From Papageorgiou, Evgeniou, and Poggio (1998)

extremely well; this test sequence was gathered with a different camera, in a different location, and in different lighting conditions than our training data.

5 Object Recognition in IT cortex

5.1 The learning-from-examples framework almost implies a view-based approach to object recognition

As we mentioned in the introduction, ten years ago a learning approach to object recognition – based on Gaussian Radial Basis Functions – suggested a view-based approach to recognition Poggio and Edelman (1990). Regularization networks store a number of examples in the hidden nodes and compare the current input to each of those store examples in parallel. Instead of having an explicit 3D model of the object we wish to recognize, we instead have a number of 2D examples of what the object looks like and we compare a current view against each of the stored examples. Different simulations with artificial (Poggio and Edelman 1990) and real “wire-frame” objects (Brunelli and Poggio 1991) and also with images of faces (Beymer 1993; Romano 1993) showed that a view-based scheme of this type can be made to work well.

It was not surprising that one of the first questions we asked was whether a similar approach may be used by our brain. As Poggio and Girosi (1989) and Poggio (1990) argued, networks that learn from examples have an obvious appeal from the point of view of neural mechanisms and available neural data. In a certain sense, networks like Gaussian Radial Basis functions are an extension of a very simple device: look-up tables. The idea of replacing computation with memory is appealing, especially from the point of view of biological and evolutionary plausibility. Interpolating or approximating memory devices such as RBF avoid many of the criticisms of pure look-up table theories. It was therefore natural for our group to try to see how far we could push this type of brain theories.

Somewhat surprisingly to us, over the last ten years many psychophysical experiments (for the first such work see Bülthoff and Edelman (1992)) have supported the example-based and view-based schemes that we suggested as one of the mechanisms of object recognition. More recent physiological experiments have provided a suggestive glimpse on how neurons in IT cortex (the area of the brain responsible for object recognition) may represent objects. The experimental results seem again to agree (so far!) to a surprising extent with the model (Logothetis, Pauls, and Poggio 1995). We are now developing a more detailed model of the circuitry and the mechanisms underlying the properties of the view-tuned units of the model (Riesenhuber and Poggio 1998).

5.2 View-based Model

Here we will review briefly our model and the physiological evidence for it. Figure 12 shows our basic module for object recognition. Classification of a visual stimulus is accomplished by a network of units. Each unit is broadly tuned to a particular view of the object. We refer to this optimal view as the center of the unit and to the unit as a *view-tuned unit.* One can think of it as a template to which the input is compared. The unit is maximally excited when the stimulus exactly matches its template but also responds proportionately less to similar stimuli. The weighted sum of activities of all the units represents the output of the network. The simplest recognition scheme of this type is the Gaussian RBF network (see equation 3): each center stores a sample view of the object and acts as a unit with a Gaussian-like recognition field around that view. The unit performs an operation that could be described as "blurred" template matching. At the output of the network the activities of the various units are combined with appropriate weights, found during the learning stage.

Consider how the network "learns" to recognize views of the object shown in figure 13. In this simplified and non-biological example the inputs of the network are the x, y positions of the vertices of the wireframe object in the image. Four training views are used. After training, the network consists of four units, each one tuned to one of the four views as in figure 13. The weights of the output connections are determined by minimizing misclassification errors on the four views and using as negative examples views of other similar objects ("distractors").

The figure shows the tuning of the four units for images of the "correct" object. The tuning is broad and centered on the center of the unit, that is the training view. Somewhat surprisingly, the tuning is also quite selective: the thinly dotted line shows the average response of each of the unit to 300 similar distractors (paperclips generated by the same mechanisms as the target; for further details about the generation of paperclips see Edelman and Bülthoff (1992)). Even the maximum response to the best distractor is in this case

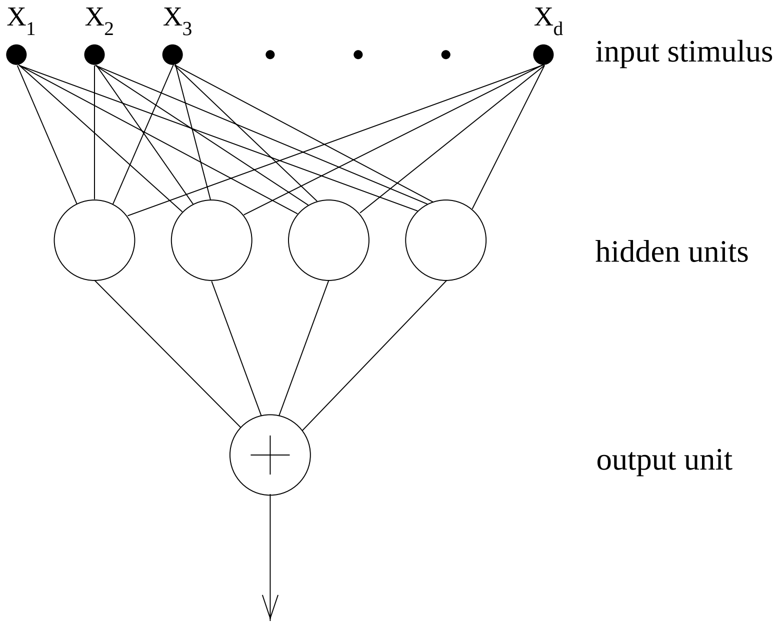

Figure 12: A Gaussian RBF network with four *view-tuned* units which, after training, are each tuned to one of the four training views shown in the next figure. The resulting tuning curve of each of the unit is also in the next figure. The units are view-dependent and selective, relative to distractor objects of the same type.

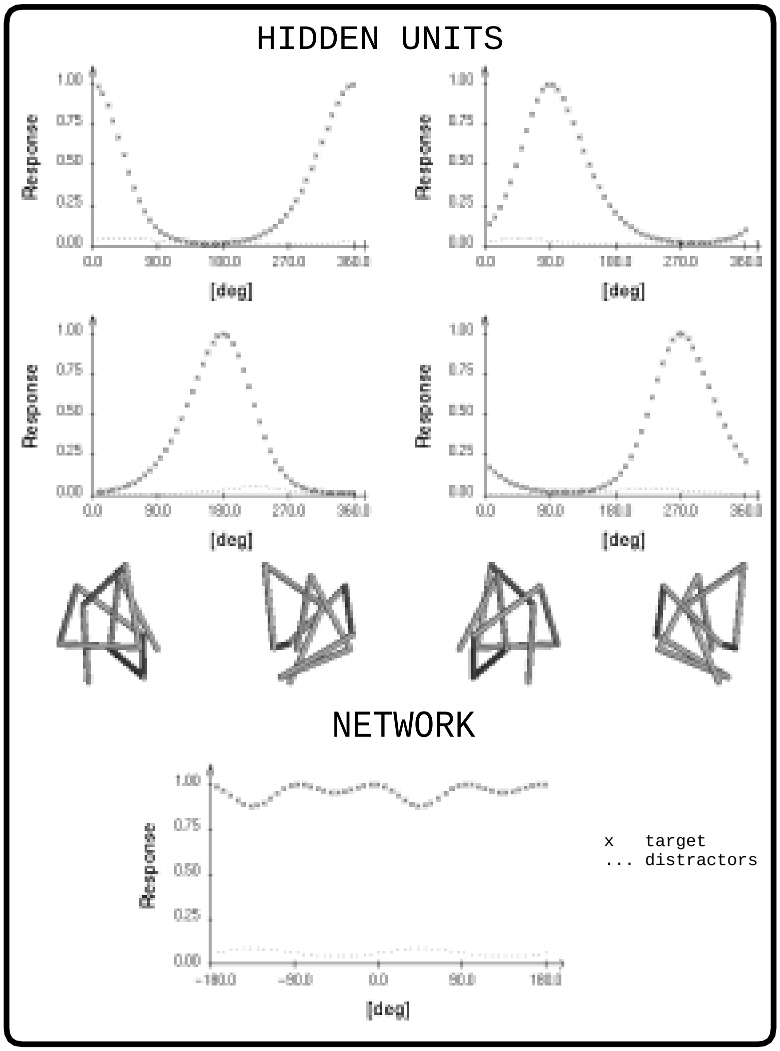

Figure 13: *Tuning of each of the four hidden units of the network of the previous figure for images of the "correct" 3D objects. The tuning is broad and selective: the dotted lines indicate the average response to* 300 *distractor objects of the same type. The bottom graphs show the tuning of the output of the network of the previous figure after learning (that is computation of the weights c): it is view-invariant and object specific. Again the dotted curve indicates the average response of the network to the same* 300 *distractors. From Vetter and Poggio, unpublished.*

always less than the response to the optimal view. The output of the network, being a linear combination of the activities of the four units, is essentially view-invariant and still very selective. Notice that each center can be regarded as the *conjunction* of all the features represented: the Gaussian can be in fact decomposed into the product of one-dimensional Gaussians, each for each input component, that is for each feature. The activity of the unit measures the global similarity of the input vector to the center: for optimal tuning all features have to be closed to the optimum value. Even the mismatch of a single component of the template may set to zero the activity of the unit. Thus the rough rule implemented by a view-tuned unit is the conjunction of a set of predicates, one for each input feature, measuring the match with the template. On the other hand the output of the network is performing an operation more similar to the "OR" of the outputs of the units.

This example is clearly a caricature of a view-based recognition module but it helps making the main points of the argument. Of course, biologically plausible features are different from the coordinates of the corners used by the toy network described above. We (Bricolo, Poggio, and Logothetis 1997; Riesenhuber and Poggio 1998) have recently performed simulations of a biologically more plausible network in which we first filter the image through a bank of directional filters of various orders and scale, similar to V1 neurons (cells in the part of the brain through which the visual information first passes). Before describing in more detail the model work on the circuitry underlying the properties of view-tuned cells, we will summarize the physiological findings (Logothetis, Pauls and Poggio, 1995; Logothetis and Pauls, 1995).

5.3 Experimental Evidence

Two monkeys were trained to recognize computer-rendered objects irrespective of position or orientation. The monkeys first were allowed to inspect an object, the *target*, presented from a given viewpoint, and subsequently were tested for recognizing views of the same object generated by rotations. In some experiments the animals were tested for recognizing views around either the vertical or the horizontal axis, and in some others the animals were tested for views around all three axes. The images were presented sequentially, with the target views dispersed among a large number of other objects, the *distractors*. Two levers were attached to the front panel of the chair, and reinforcement was contingent upon pressing the right lever each time the target was presented. Pressing the left lever was required upon presentation of a distractor. Correct responses were rewarded with fruit-juice.

An observation period began with the presentation of a small fixation spot. Successful fixation was followed by the *learning phase*, whereby the target was inspected for 2 seconds from one viewpoint, the training view. The learning phase was followed by a short fixation period after which the *testing phase* started. Each testing phase consisted of up to 10 trials, in each of which the test stimulus, a shaded, static view of either the target or a distractor was presented.

A total of 970 IT cells were recorded from two monkeys during combined psychophysical-electrophysiological experiments. Logothetis and coworkers found a significant number of units that showed a remarkable selectivity for individual views of wire objects that the monkey was trained to recognize.

Figure 14 shows the responses of three units that were found to respond selectively to four different views of an wire object (Wire 71). The animal had been exposed repeatedly to this object, and its psychophysical performance remains above 95% for all tested views, as can be seen in the lower plot of figure 14. Notice that one of the 3 neurons is tuned to a view and its mirror image, consistently with other theoretical and psychophysical work. The figure is surprisingly similar to figure 13 showing the response of the view-tuned hidden units of the model of figure 12.

A small percentage of cells (8 out of 773) responded to wire-like objects presented from any viewpoint, thereby showing view-invariant response characteristics, superficially similarly to the output unit of the model of figure 12. An example of such a neuron is shown in figure 15. The upper plot shows the monkey's hit rate and the middle plot the neuron's average spike rate. The cell fires with a rate of about 40Hz for all target's views. The lower plot shows the responses of the same cell to 120 distractors. With four exceptions activity was uniformly low for all distractor objects presented. In all cases, even the best response to a distractor, however, remains about one half of the worst response to a target view. This neuron seems to behave as the output of the model of figure 12. 71 out of the 773 (9%) analyzed cells showed view selective responses similar to those illustrated in the two preceding figures. In their majority, the rest of the neurons were visually active when plotted with other simple or complex stimuli, including faces.

The main finding of this study is that there are neurons in IT cortex with properties intriguingly similar to the "cartoon" model of figure 12, which is itself supported by psychophysical experiments in humans and primates. Several neurons showed a remarkable selectivity for specific views of a computer-rendered object that the monkey had learned to recognize. A much smaller number of neurons were object-specific but view-invariant, as expected in a network in which "complex"-like view-invariant cells are fed by view-centered "simple"-like units. Furthermore, we believe that our results reflect experience dependent plasticity in IT neurons and quite possibly also much earlier in the visual pathway. First, the neurons we found responded selectively to novel visual objects that the monkey had learned to recognize during the training. None of these objects had any prior meaning to the animal and none of them resembled anything familiar in the monkey's environment. In addition, no selective responses were ever encountered for views that the animal systematically failed to recognize. Thus it seems that neurons in this area can develop a complex selectivity as a result of training in the recognition of specific objects. Notice that view-tuning was observed only for those views that the monkey could recognize.

A back-of-the-envelope extrapolation of the available data suggests an estimate of the number of cells whose tuning was determined by the training. In the region of IT from which recording were made, which contains around ten

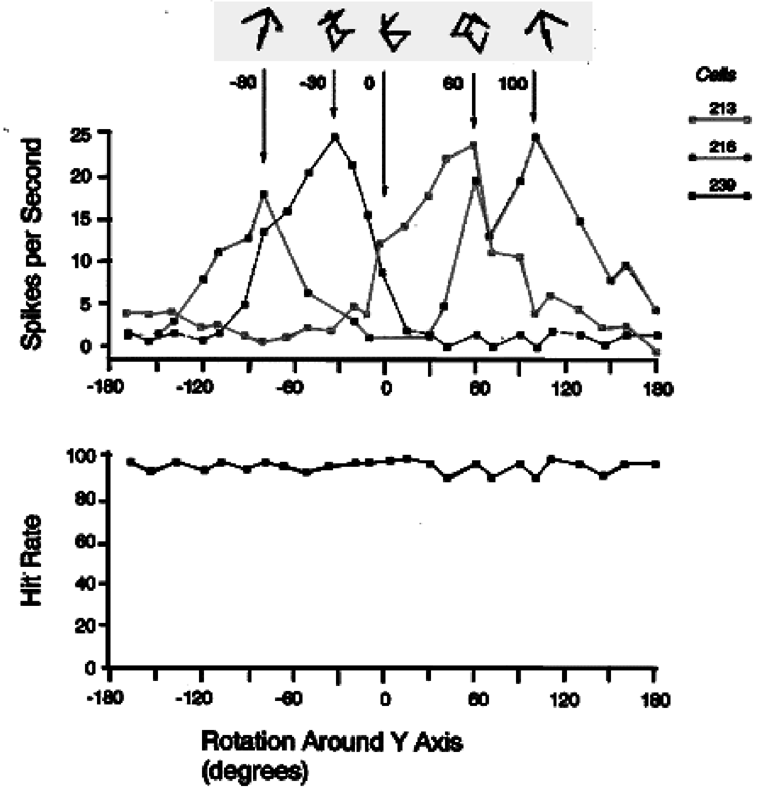

Figure 14: The top graph shows the activity of three units in IT cortex, as a function of the angle of the stimulus view. The three neurons are tuned to four different views of the same object, in a similar way to the units of the model of figure 12 and figure 13. One of the units shows two peaks for two mirror symmetric views. The neurons firing rate was significantly lower for all distractors (not shown here). The bottom graph represents the almost perfect, view-invariant behavioral performance of the monkey for this particular object to which he was extensively trained (from Logothetis and Pauls, unpublished, 1995).

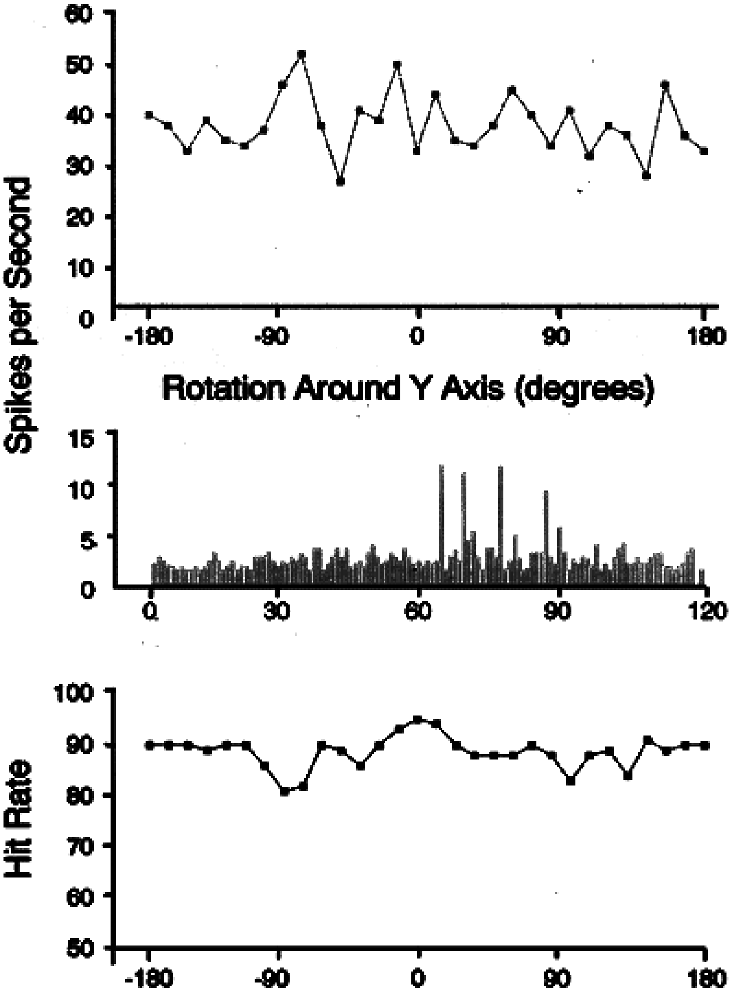

Figure 15: The monkey performed quite well on this particular wire after extensive training (bottom graph). A neuron in IT was found that shows a view-invariant response, with about 30 spikes/sec to any view of the wire object (top). The response of the cell to any of the 120 distractors is lower as shown in the middle graph (from Logothetis and Pauls, unpublished). This is similar to the output unit of the model of figure 12, see figure 13

million neurons, we estimate that for each of the about twelve objects that the monkeys had learned to recognize there were, at the time of the recordings, a few hundred view-tuned cells and in the order of 40 or so view-invariant cells.

5.4 A New Model

Models like the one of figure 12 leave open the issue of the mechanisms and circuitry underlying the properties of the view-tuned cells, from their view tuning to their invariance to image-based transformations such as scaling and translation. In fact, the invariance of the view-tuned neurons to image-plane transformation and to changes in illumination has been tested experimentally by Logothetis, Pauls, and Poggio (1995) who report an average rotation invariance over 30 degrees, translation invariance over 2 degrees, and size invariance of up to 1 octave around the training view.

These recent data put in sharp focus and in quantitative terms the question of the circuitry underlying the properties of the view-tuned cells. The key problem is to explain in terms of biologically plausible mechanisms their viewpoint invariance obtained from just one object view, which arises from a combination of selectivity to a specific object and tolerance to viewpoint changes.

Riesenhuber and Poggio (1998) have described a model that conforms to the main anatomical and physiological constraints, reproduces all the data obtained by Logothetis et al. and makes several predictions for experiments on a subpopulation of IT cells. A key component of the model is a cortical mechanism that can be used to either provide the sum of several afferents to a cell or to enable only the strongest one. The model explains the receptive field properties found in the experiment based on a simple hierarchical feedforward model. The structure of the model reflects the idea that invariance and specificity must be built up through separate mechanisms. Figure 16 shows connections to “invariance” units in green and to “specificity” units in blue.

This new model is an expansion of the previous model to include non-linear maximum (MAX) operation (similar to Nearest Neighbor classication) to allow a high degree of invariance. This new model in simulations shows agreement with several physiological experiements from different labs. In particular, figure 17 shows the predictionsof the model in comparison with experimental data.

6 Conclusions

The diagram at the beginning of this article in figure 1 shows our research process as a continuous loop. Because of the linearity of the print medium, we have mainly been able to show how theory has inspired applications and how their success have influenced research in neuroscience.

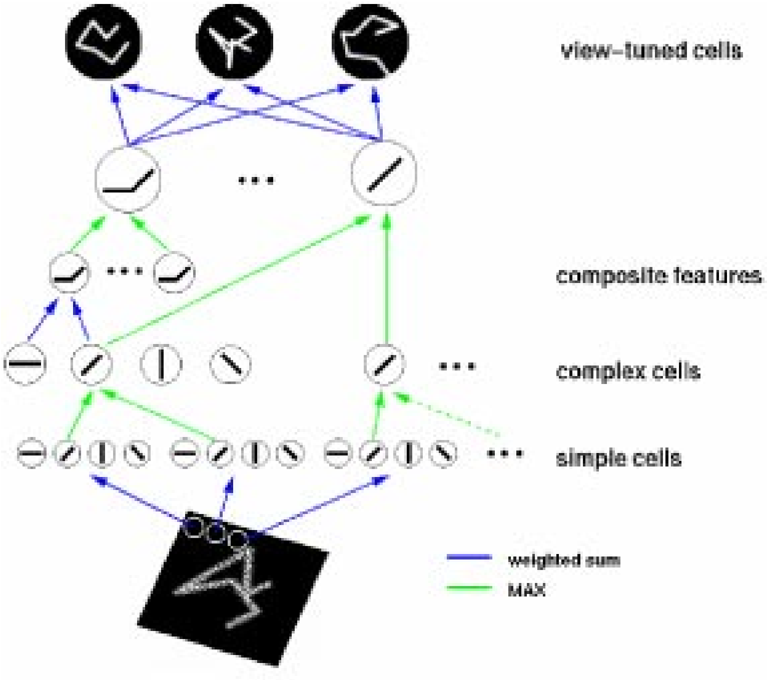

Figure 16: Model to explain receptive field properties of the view-tuned units of figure 12 found in experiments (from Riesenhuber and Poggio, in preparation)

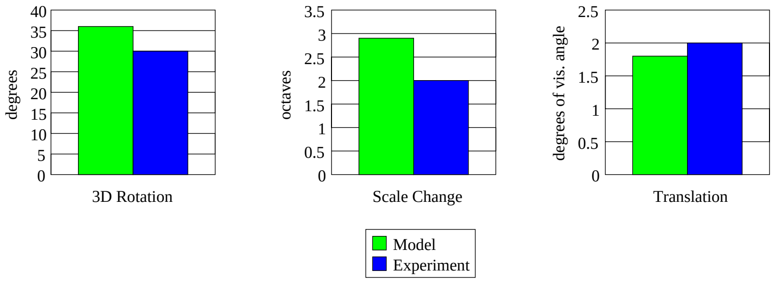

Figure 17: Comparison of theoretical model and experimental data

However, the flows of ideas also go the other way. The results shown for pedestrian detection of the influence of the number of features on the final classification results begs a very important theoretical question: “What is the optimal feature set and how can we find it?” Furthermore, applications seem to beat the theoretical upper bounds by quite a large margin. This is especially true when one considers that the data-independent bounds derived for structural risk minimization should be overly optimistic since, as actually implemented, support-vector machines require data-dependent bounds (which should be worse). So, the applications pose another theoretical question: “Can we find good data-dependent bounds for machine learning algorithms?”

The neuroscience work also raises theoretical and application questions. For example, the model depicted in figure 16 is inspired by invariances found in neurons. So the next step is to try to use such results to direct new theories which might explain why such a model is a good one and use similar banks of filters as feature detectors in applications.

The supervised learning learning paradigm outlined here can be applied to other domains as well, beyond the area of vision. For instance, over the years we have applied it to computer graphics. By analogy to the view-based paradigm for computer vision we were led to the paradigm of image based rendering which is just now becoming an important research direction in the graphics community (Librande 1992; Beymer and Poggio 1996; Ezzat and Poggio 1998). Other applications of our learning techniques have been in the domain of time series and finance (see for instance Hutchinson, Lo, and Poggio (1994)), in control, and in search engines.

Despite a number of interesting and useful applications, it is clear that the problem of building machines that learn from experience and the problem of understanding how our brain learns are still wide open. Most of the really challenging questions are unsolved. There are still gaps between theory and applications and between machine learning and biological learning. Such comparisons raise a number of interesting questions including:

- Why is there a large difference between the number of examples a machine learning algorithm needs (usually thousands) and the number of examples the human brain requires (just a few)?

- What is the best way of naturally incorporating unlabeled examples into the supervised learning framework?

- Can supervised learning methods be used to attack or solve other types of learning problems such as reinforcement learning and unsupervised learning?

- To what extent can supervised learning explain the adaptive systems of the brain?

We hope that the work we described represents a few small steps in the right direction, in addition to providing a lot of fun for the mathematicians, the engineers, and the neuroscientists who are involved.

References

Beymer, D. and T. Poggio (1996, June). Image representations for visual learning. *Science 272*(5270), 1905–1909.

Beymer, D. J. (1993). Face recognition under varying pose. A.I. Memo No. 1461, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Breiman, L. (1993, May). Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transaction on Information Theory 39*(3), 999–1013.

Bricolo, E., T. Poggio, and N. Logothetis (1997). 3d object recognition: A model of view-tuned neuron. In M. M. M.I. Jordan and T. Petsche (Eds.), *Advances in Neural information processings systems 3*, Cambridge, MA. M.I.T. Press.

Broomhead, D. and D. Lowe (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems 2*, 321–355.

Brunelli, R. and T. Poggio (1991). Hyberbf networks for real object recognition. In *Proceedings IJCAI*, Sydney, Australia.

Bülthoff, H. H. and S. Edelman (1992). Psychophysical support for a 2-D view interpolation theory of object recognition. *Proceedings of the National Academy of Science 89*, 60–64.

Edelman, S. and H. H. Bülthoff (1992). Orientation dependence in the recognition of familiar and novel views of 3D objects. *Vision Research 32*, 2385–2400.

Evgeniou, T., M. Pontil, and T. Poggio (1999, March). A review of a unified framework for regularization networks and support vector machines. A.i. memo, MIT Artificial Intelligence Lab.

Ezzat, T. and T. Poggio (1998). Miketalk: A talking facial display based on morphing visemes. In *Proceedings of the Computer Animation Conference*, Philadelphia. Morgan Kaufman.

Friedman, J. and W. Stuetzle (1981). Projection pursuit regression. *Journal of the American Statistical Association 76*(376), 817–823.

Girosi, F. (1998). An equivalence between sparse approximation and Support Vector Machines. *Neural Computation 10*(6), 1455–1480.

Girosi, F., M. Jones, and T. Poggio (1993). Priors, stabilizers and basis functions: From regularization to radial, tensor and additive splines. A.I. Memo No. 1430, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Girosi, F., M. Jones, and T. Poggio (1995). Regularization theory and neural networks architectures. *Neural Computation 7*, 219–269.

Hutchinson, J., A. Lo, and T. Poggio (1994, July). A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance XLIX*(3), 851–889.

Librande, S. (1992, September). Example-based character drawing. Master's thesis, M.S., Media Arts and Science Section, School of Architecture and Planning, Massachusetts Institute of Technology, Cambridge, MA.

Logothetis, N., J. Pauls, and T. Poggio (1995). Shape Representation in the Inferior Temporal Cortex of Monkeys. *Current Biology 5*(5), 552–563.

Micchelli, C. A. (1986). Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximation 2*, 11–22.

Mitchell, T. (1997). *Machine Learning*. Boston: McGraw-Hill.

Oren, M., C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio (1997, June 16–20). Pedestrian detection using wavelet templates. In *Proc. Computer Vision and Pattern Recognition*, Puerto Rico, pp. 193–199.

Osuna, E., R. Freund, and F. Girosi (1997, June 16–20). Training support vector machines: an application to face detection. In *Proc. Computer Vision and Pattern Recognition*, Puerto Rico.

Papageorgiou, C. (1997). Object and Pattern Detection in Video Sequences. Master's thesis, MIT.

Papageorgiou, C., T. Evgeniou, and T. Poggio (1998, October). A trainable pedestrian detection system. In *Proceedings of Intelligent Vehicles*, Stuttgart, Germany, pp. 241–246.

Papageorgiou, C., M. Oren, and T. Poggio (1998, January). A general framework for object detection. In *Proceedings of the International Conference on Computer Vision*, Bombay, India.

Poggio, T. (1990). 3D object recognition: on a result by Basri and Ullman. Technical Report # 9005–03, IRST, Povo, Italy.

Poggio, T. and S. Edelman (1990). A network that learns to recognize 3D objects. *Nature 343*, 263–266.

Poggio, T. and F. Girosi (1989). A theory of networks for approximation and learning. A.I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Poggio, T. and F. Girosi (1990a, September). Networks for approximation and learning. *Proceedings of the IEEE 78*(9), 1481–1497.

Poggio, T. and F. Girosi (1990b). Regularization algorithms for learning that are equivalent to multilayer networks. *Science 247*, 978–982.

Powell, M. J. D. (1987). Radial basis functions for multivariable interpolation: a review. In J. C. Mason and M. G. Cox (Eds.), *Algorithms for Approximation*. Oxford: Clarendon Press.

Riesenhuber, M. and T. Poggio (1998). Modeling invariances in inferotemporal cell tuning. A.I. Memo No. 1629, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Romano, R. (1993). Real-time face verification. Master's thesis, Massachusetts Institute of Technology.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory.* New York: Springer.

Wahba, G. (1990). *Splines Models for Observational Data.* Philadelphia: Series in Applied Mathematics, Vol. 59, SIAM.