Incorporating Prior Information in Machine Learning by Creating Virtual Examples

P. Niyogi, F. Girosi, and T. Poggio
MIT Center for Biological and Computational Learning
Cambridge, MA 02139.

September 2, 1998

Abstract

One of the key problems in supervised learning is the insufficient size of the training set. The natural way for an intelligent learner to counter this problem and successfully generalize is to exploit prior information that may be available about the domain or that can be learned from prototypical examples. We discuss the notion of using prior knowledge by creating *virtual examples* and thereby expanding the effective training set size. We show that in some contexts, this idea is mathematically equivalent to incorporating the prior knowledge as a regularizer, suggesting that the strategy is well-motivated. The process of creating virtual examples in real world pattern recognition tasks is highly non-trivial. We provide demonstrative examples from object recognition and speech recognition to illustrate the idea.

1 Learning from Examples

Recently, machine learning techniques have become increasingly popular as an alternative to knowledge-based approaches to artificial intelligence problems in a variety of fields. The hope is that automatic learning from examples will eliminate the need for laborious handcrafting of domain-specific knowledge about the task at hand. However, analyses of the complexity of learning problems suggest that this hope might be overly optimistic — often the number of examples needed to solve the problem might be prohibitive. Clearly, a middle ground is needed and a useful direction of research is the study of how to incorporate prior world knowledge of the task within a learning from examples framework.

The current paper deals with this subject. We first begin by providing some background about how the problem of learning from examples is usually formulated. In the next section, we discuss briefly the complexity of the learning problem and why, in the absence of any prior knowledge, one might require a large number of examples to learn well. In section 3, we introduce the idea of virtual examples, i.e., creating additional examples from the current set of examples by utitilizing specific knowledge about the task at hand. While the overall framework is similar to learning from hints (Abu-Mostafa,1995), our emphasis in this paper is to describe some specific non-trivial transformations that allow us to create virtual examples for real world pattern recognition problems. We first show in section 4 that in certain function learning contexts, the framework of virtual examples is equivalent to imposing prior knowledge as a regularizer. Thus, the idea of virtual examples can be more than an ad hoc strategy. We then discuss in section 5, some specific examples from computer vision and speech recognition. Finally, we conclude by reiterating some of our main points in section 6.

1.1 Background: Learning as Function Approximation

The problem of learning from examples can be usefully modeled as trying to approximate some unknown target function f from (x, y) pairs that are consistent with this function (modulo noise). The target function f belongs to some *target class* of functions denoted by $\mathcal{F}$. The learner has access to a data set consisting of (say) n (x, y) pairs $((x_i, y_i) : i = 1, \ldots, n)$ and picks a function h chosen from some *hypothesis class* $\mathcal{H}$ on the basis of this data set. The hope is that if "enough" examples are drawn, the learner's hypothesis will be sufficiently close to the target resulting in successful generalization to novel unlabelled examples that the learner might encounter.

Numerous problems in pattern recognition, speech, vision, handwriting, finance, robotics, etc. can be cast within this framework and research typically focuses on different kinds of hypothesis classes ($\mathcal{H}$) and different ways of choosing an optimal function in this class (training algorithms). Thus, multilayer perceptrons (Rumelhart, Mclelland and Hinton, 1986), radial basis function networks (Poggio and Girosi, 1990; Moody and Darken, 1988), decision trees (Breiman etal, 1984), all correspond to different choices of hypothesis classes on which popular learning machines have been based. Similarly, different kinds of gradient descent schemes from backpropagation to the EM algorithm correspond to ways of choosing an optimal function from such a class given a finite data set. By varying the choices of hypothesis classes and training algorithms, a profusion of learning paradigms have emerged. The most significant issue of interest in each of these learning paradigms is how they generalize to new unseen data. In the next section, we discuss the factors upon which the generalization performance of a learning machine depends.

2 Prior Information and the Problem of Sample Complexity

In any learning from examples system, the number of examples (l) that needs to be collected for successful generalization is a key issue. This *sample complexity* is typically characterized by the theory of Vapnik and Chervonenkis (1982,1995) that describes the general laws that all probabilistically based learning machines have to obey. It turns out that if the learner picks a hypothesis ($\hat{h} \in \mathcal{H}$) on the basis of the example set, then the number of examples it needs in order to generalize well is of the order of $\sqrt{\frac{VC(\mathcal{H})}{l}}$. Here $VC(\mathcal{H})$ is the VC-dimension of the class $\mathcal{H}$ — a combinatorial measure of the complexity of the hypothesis class. Roughly speaking, the VC dimension (see Vapnik, 1982 for further details) is a measure of how many different kinds of functions there are in $\mathcal{H}$. For example, if $\mathcal{H}$ were the parametric class of univariate polynomials of degree n, its VC dimension[1] is $n+1$. In

[1] While in this case, the VC dimension is related in a simple way to the number of parameters, this need not be true in general. One can think of classes with many parameters having a small VC dimension and vice versa. Thus the VC dimension is a better and more direct measure of learning complexity than simply the number of parameters.

general, large or complex hypothesis classes that can accomodate many different data sets would have a higher VC dimension than smaller, restricted hypothesis classes. Thus we see that the number of examples needed is proportional to the VC-dimension, and in this sense, to the effective size of the hypothesis class. Consequently, it is in our interest to use small hypothesis classes in learning machines.

However, using a small hypothesis class is not enough. Recall that the target function f belongs to $\mathcal{F}$ and if our hypothesis class $\mathcal{H}$ is too small, then, even if we choose the best function in it, the distance from the target (generalization error) might be too high. To appreciate this point better, let us consider a situation of learning using neural networks in a least-squares setting. Recall that ideally, we would like to "learn" the target function that is given by the following (the expectation is with respect to the true probability distribution generating the data):

$$f_0 \equiv \arg\min_{f \in \mathcal{F}} E[(y - f(x))^2]$$

However, in practice, we don't know the true distribution and so cannot compute the true expectation; nor do we typically minimize over the class $\mathcal{F}$. For example, consider the typical situation if we were using neural networks to learn the function f_0. We draw a finite data set (x, y pairs), construct an empirical approximation to the objective function and then minimize this over a class of neural networks with a finite number of parameters. If we collected l data points and minimized over a neural network with n nodes in its hidden layer (say), we are in effect computing the following function $\hat{f}_{n,l}$:

$$\hat{f}_{n,l} = \arg\min_{h \in \mathcal{H}_n} E_{emp}[(y - h(x))^2] \equiv \min_{h \in \mathcal{H}_n} \frac{1}{l} \sum_{i=1}^{l} (y_i - h(x_i))^2$$

Thus, when we attempt to learn the function f_0 using a finite amount of data (l points) and a hypothesis class with a finite number of parameters ($\mathcal{H}_n$) then, the function we obtain in practice is given by $\hat{f}_{n,l}$. This is the function we use to predict future, unknown values and naturally, we would like to know how good this function is, i.e., how far this function is from the true target. In general, one can show that the generalization error ($\| f_0 - \hat{f}_{n,l} \|$) can be decomposed into an approximation component and an estimation component, i.e.,

$$\| f_0 - \hat{f}_{n,l} \| \leq e_{app}(n) + e_{est}(\frac{VC(\mathcal{H}_n)}{l})$$

The approximation error $e_{app}(n)$ is due to the finite size of the hypothesis class. As the number of hidden nodes, n, increases, the representational power of the hypothesis class increases and the approximation error goes to zero. The estimation error $e_{est}(\frac{VC(\mathcal{H}_n)}{l})$ is due to the finite amount of data that is available to the learner. It is a monotonically decreasing function and depends upon the VC dimension of the hypothesis class $\mathcal{H}_n$ and the amount of data (l). As the number of hidden nodes increases, the VC dimension of $\mathcal{H}_n$ increases and consequently the estimation error increases as well (keeping the data fixed). Thus to make the approximation error small, we need large sized networks (n large); to make the estimation error small, we need small sized networks (n small). This trade-off between the approximation error and estimation error arises in all learning paradigms and has been investigated for a number of different hypothesis classes ranging from multilayer perceptrons (Barron, 1994) to radial basis functions (Niyogi and Girosi, 1996). The following theorem states a canonical result for radial basis functions and fig. 1 below describes the generalization error surface as a function of the number of parameters and the number of data.

Theorem 1 (Niyogi and Girosi,1996) *Let H_n be the class of Gaussian Radial Basis Function networks with k input nodes and n hidden nodes, i.e.,*

$$H_n = \sum_{i=1}^{n} c_i G_i(\frac{\mathbf{x}_i - \mathbf{t}_i}{\sigma_i})$$

Let f_0 be an element of the Bessel potential space[2] *$\mathcal{L}_1^m(R^k)$ of order m, with $m > k/2$ (the class $\mathcal{F}$). Assume that a data set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ has been obtained by randomly sampling the function f_0 in presence of noise, and that the noise distribution has compact support. Then, for any $0 < \delta < 1$, with probability greater than $1 - \delta$, the following bound for the generalization error*

[2] This is defined as the set of functions f that can be written as $f = \lambda * G_m$, where $*$ stands for the convolution operation, $\lambda \in L_p$ and G_m is the Bessel-Macdonald kernel, i.e., the function whose Fourier transform is:

$$\tilde{G}_m(\mathrm{s}) = \frac{1}{(1 + 4\pi^2 \|\mathrm{s}\|^2)^{m/2}}$$

holds:

$$\|f_0 - \hat{f}_{n,l}\|^2_{L^2(P)} \leq O\left(\frac{1}{n}\right) + O\left(\left[\frac{nk\ln(nl) - \ln\delta}{l}\right]^{1/2}\right) \tag{1}$$

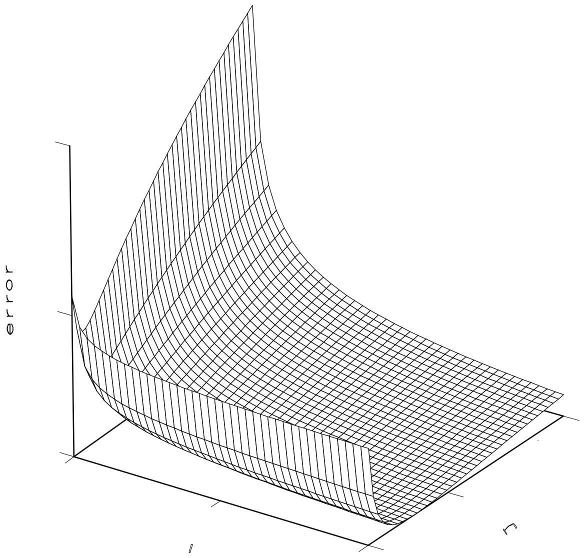

Figure 1: The generalization error, the number of examples (l) and the number of basis functions (n) as a function of each other.

What do we conclude from all this? First, that if we work with unconstrained hypothesis classes, the number of examples needed for low estimation error will almost certainly be prohibitive. On the other hand, if we work with highly constrained hypothesis classes, the approximation error will be too high for successful generalization. The only way around this dilemma is if the target class itself ($\mathcal{F}$) can be made small — but this is precisely the prior information we have about the problem being solved. Thus, if we have more prior information about the target function, it would correspond to a smaller target class and the problems with poor generalization would be ameliorated.

In essence, prior information is more than a good idea. The mathematics of generalization (from the no-free-lunch theorems of Wolpert (1995) to the statistical theory of Vapnik (1995)) all point to one thing — incorporation of prior knowledge might be the only way for learning machines to tractably generalize from finite data. One way of incorporating prior knowledge is the idea of virtual examples: utilizing prior information about the target function to generate novel examples from old data thereby enlarging our effective data set. We discuss this in the next section.

3 Virtual Examples: A Framework for Prior Information

As we have discussed above, a significant problem in learning from examples is the large amount of data (examples) needed for adequate learning. Consequently, it becomes crucial to exploit any form of prior knowledge we might have about the task at hand. A well known technique for incorporating prior information is to restrict the class of hypotheses — this would also reduce the data requirements by the Vapnik Chervonenkis theory as discussed in the previous section.

Another alternative might be to expand the set of available examples in some fashion so that the learner has access to an effectively larger set of examples resulting in more accurate learning. These additional examples, created from the existing ones by the application of prior knowledge, will be referred to as *virtual* examples (first introduced by Poggio and Vetter, 1992 and different from the notion of virtual examples introduced by Abu Mostafa (1990,1992,1993) as discussed later). We first lay the general framework for virtual examples in section 3.1. Of course, virtual examples is only one way of incorporating prior information and we will briefly review some alternate methods and their relationship to our approach in section 3.2.

At first, the idea of virtual examples might seem like an ad-hoc one but we show in section 4 the connection between the virtual examples approach and using regularization as a technique for incorporating prior information. For the example discussed, it is possible to prove that both techniques yield the same solution. The heart of the virtual examples idea involves the actual creation of the virtual examples —- this is the main focus of our paper and we discuss several substantive real world, practical demonstrations of this approach in later sections.

3.1 The General Framework

As discussed earlier, the primary goal of the learner is to approximate some unknown target function (f) from examples ((x, y) pairs) of this function. The unknown target function might be a real-valued, multivariate function (as in sec. 4), or even a characteristic funtion defined over some manifold (as in sec. 5).

In the absence of any prior information, the learner would attempt to fit a function from $\mathcal{H}$ to the data set and use it to predict future values. Suppose, however, that we have prior knowledge of a set of transformations that allow us to obtain new examples from old. For example, the target function might be invariant with respect to a particular group of transformations. A simple case is when the target function is known to be even or odd. A more complex case might be if the target function is a characteristic function defined over a manifold of 3D objects. Correspondingly, in some cases, obtaining the new examples might be easy, in other cases — as in the case for object recognition — it is quite difficult.

Thus, suppose we know some transformation T such that if $(x, f(x))$ is a valid example, then $(Tx, y_T(f(x)))$ is also a valid example. For an invariant transformation, y_T is the identity mapping. In general, of course, the relation of y_T to T depends upon the prior knowledge of the problem and might be quite complex. Then, given a set of n examples: $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ and knowledge of this transformation T, we generate the set of virtual examples: $D' = \{(x'_1, y'_1), \ldots, (x'_n, y'_n)\}$ such that $x'_i = Tx$ and $y'_i = y_T(y_i)$.

$$(x, f(x)) \xrightarrow{T, y_T} (Tx, y_T(f(x)))$$

For many interesting cases, prior knowledge of the problem might allow us to define a group of transformations $\mathcal{T}$ such that for every $T \in \mathcal{T}$, we can create new, *virtual* examples from the old data set. For example, rotations (in the image plane or in 3D) might define such a group for object recognition problems. Thus, the creation of virtual examples allows us to expand the example set and consequently move towards better generalization.

3.2 Techniques for Prior Information and Related Research

Needless to say, the idea of virtual examples is only one possible way of incorporating prior information. We discuss in this section various ways in which researchers have tried to utilize prior knowledge.

1. **Prior Knowledge in the Choice of Variables or Features:** Prior knowledge could be used in the choice of the variables or features that are used to represent the problem. Let us consider the case of object recognition. A simple form of prior knowledge is that the rotated version (in 2D) of an object still represents that object. Therefore one could think of using, as input to the network, features that are invariant under rotations in the image plane. In this case, rotation invariant recognition would be achieved with just one example. This approach is somehow limited in vision applications, because it is very difficult to find features that are invariant for "interesting" transformations. For example it does not seem likely that one can find features of face images that are invariant with respect to rotation in 3D space, apart from "trivial" ones such as the colour of the person's hair etc. (for more details on the possibilities of such an approach, see Mundy et al, 1992).

   Another kind of prior knowledge could be that certain features always appear in conjunction (or disjunction), or certain variables are always linked together in a certain form. In this case one could explicitly add these new variables to the set of original variables, making the learning task much easier. For example, in robotics it is known that for certain mechanical systems, the relation between torques and state variables is represented by certain combinations of trigonometric functions. Therefore, explicitly adding sine and cosine transformation of the state space variables usually makes the problem much easier to solve. This technique is also not uncommon in statistics, where often new variables are created by means of nonlinear transformation of the original ones.

2. **Prior Knowledge in the Learning Technique:** Another way to incorporate prior knowledge is to embed it in the learning technique. Examples of this are the recent **transformation distance** technique introduced by Simard, Le Cun and Denker (1993). The idea underlying this technique is the following: suppose a pattern classification

problem has to be solved, and we know that the outcome of the classification scheme should be invariant with respect to a certain transformation $R(\mathbf{w})$, where $\mathbf{w}$ is a set of parameters (for example the rotation angle in the image plane for object recognition). This means that for every input pattern $\mathbf{x}$ there is a manifold $S\mathbf{x}$ on which the output should be constant. Therefore, if we desire to use a classification technique such as Nearest Neighbors, that is based on a notion of distance, we should use as distance between two patterns $\mathbf{x}$ and $\mathbf{z}$ not the Euclidean distance between them, but the Euclidean distance between the manifolds $S(\mathbf{x})$ and $S(\mathbf{z})$. This quantity cannot be computed analytically, in general, but Simard, Le Cun and Denker (1993) show how to estimate it using a local approximation of the manifold $S(\mathbf{x})$ by its tangent plane that can be experimentally computed. In this case the prior knowledge has been embedded in the definition of distance, and therefore in the learning technique, rather than in the choice of the variables, as described above.

Another case in which prior knowledge is embedded in the learning technique is regularization theory, a set of mathematical tools introduced by Tikhonov in order to deal with ill-posed problems (Tikhonov, 1963; Tikhonov and Arsenin, 1977; Morozov, 1984; Bertero, 1986; Wahba, 1990; Poggio and Girosi, 1990; Girosi, Jones and Poggio, 1995). In regularization theory an ill-posed problem is transformed into a well-posed one using some prior knowledge. The most common form of prior knowledge is smoothness, whose role in the theory of learning from examples has been investigated at length by Poggio and Girosi (1990). However, other forms of prior knowledge can be used in the framework of regularization theory. This topic has been investigated by Verri and Poggio (1986), who gave sufficient conditions for a constraint to be embedded in the regularization framework. Examples of the prior knowledge they considered include monotonicity, convexity and positivity.

3. **Generating New Examples with Prior Knowledge:** Another form of utilising prior knowledge for learning is the idea of generating new examples from the existing data set. This is the idea of virtual examples (from Poggio and Vetter, 1992) that we consider in this paper. An example of a similar technique can be found in the work of Pomerleau (1989, 1991) on ALVINN, an autonomous navigation vehicle that learns to drive on a highway. The system consists of a camera mounted

on a vehicle, and a neural network that takes that the image of the road as an input and produces as output a set of steering commands. The examples are acquired by recording the actions of a human driver. Since humans are very good at keeping the vehicle in the right lane, the images of the road look all alike, and there are no examples of what action to take if the vehicle is in an "unusual attitude", that is, too far to the right or to the left. Therefore the network is not able to give correct answers if it finds itself in these kinds of situations, of which it has no examples. Pomerleau used prior knowledge on the geometry of the problem in order to create examples of what to do in the case of unusual attitudes. Knowing the location of the camera, with respect to the vehicle, and based on examples belonging to the data set created by the human driver, he was able to create images of what the road would look like if the vehicle were in an unusual attitude, say too close to the centerline. Given these new images and the corresponding locations of the vehicle he computed what the steering command should be for each one of them, creating a whole new set of images, now containing many examples of unusual attitudes, and allowing the system to achieve excellent performance.

4. **Incorporating Prior Knowledge as Hints:** Another technique is the one proposed by Abu-Mostafa (1990, 1992, 1993). Here we list very briefly the main points of his concept of *hints*. The approach overlaps to a good extent but not completely with our own ideas of virtual examples.

   Consider

   (a) a function f to be learned with domain, X, and range, Y;

   (b) the hypothesis g provided by the learning process, say by a Regularization Network approximation of f;

   (c) the functional $E(g, f)$ measuring the error.

   Then a *hint* H_m is a test that f must satisfy. One generates one example of the hint and measures e_m, the amount of error of g on that example (if the hint is that f is odd then one chooses an x and uses $e_m = (g(\mathbf{x}) + g(-\mathbf{x}))^2$). The total disagreement between g and H_m is then $E_m = E(e_m)$

   Here are some examples of hints

- Invariance hint: $f(\mathbf{x}) = f(\mathbf{x}')$ for certain examples $(\mathbf{x}, \mathbf{x}')$. The associated error can be $e_m = (g(\mathbf{x}) - g(\mathbf{x}'))^2$
- Monotonicity hint: $f(z) < f(z')$ for certain examples (z, z') for which $z \leq z'$. Then the associated error can be $e_m = (g(z) - g(z'))^2$ if $g(z) > g(z')$ and $e_m = 0$ otherwise.
- Example hint: the set of examples of f can be treated as a hint, H_0

Abu-Mostafa (1992, 1993) describes how to represent hints by virtual examples. It is important for us to distinguish our notion of virtual examples from that of Abu-Mostafa. For Abu-Mostafa, a virtual example is typically a pair, (x, x') that are related in some way by the hint. Minimization is then done over all virtual examples. On the other hand, Abu-Mostafa also introduces the notion of *duplicate* examples. These are (x, y) pairs in the traditional sense that are somehow created by knowledge of the hint. They are often associated with invariant sets and are essentially the same as our virtual examples. While Abu-Mostafa focuses on the learning mechanism (a kind of adaptive minimization scheme) to use the hint once it has been represented by the creation of virtual examples (or duplicate examples), our focus here is on the actual creation of the virtual examples for some non-trivial learning problems.

4 Virtual Examples and Regularization

We begin by showing that the idea of virtual examples can lead to a solution that is identical to that obtained by incorporating the prior knowledge as a regularizer. Related results have also been obtained by Leen (1995) and Bishop (1995).

4.1 Regularization Theory and RBF

Suppose that the set $D = \{(\mathbf{x}_i, y_i) \in R^d \times R\}_{i=1}^N$ is a random, noisy sample of some multivariate function h. The problem of recovering the function h from the data D is ill-posed, and can be formulated in the framework of regularization theory (Tikhonov, 1963; Wahba, 1990; Poggio and Girosi, 1990). In this framework the solution is found by minimizing a functional of the form:

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

where λ is a positive number that is usually called the *regularization parameter* and $\phi[f]$ is a cost functional that constrains the space of possible solutions according to some form of prior knowledge. The most common form of prior knowledge is *smoothness*, that, in words, ensures that if two inputs are close the two corresponding outputs are also close. We consider here a very general class of rotation invariant smoothness functionals (Girosi, Jones and Poggio, 1995), defined as

$$\phi[f] = \int_{R^d} d\mathbf{s} \ \frac{|\tilde{f}(\mathbf{s})|^2}{\tilde{G}(\mathbf{s})}$$

where $\tilde{}$ indicates the Fourier transform, $\tilde{G}$ is some positive radial function that tends to zero as $\|\mathbf{s}\| \rightarrow \infty$ (so that $\frac{1}{\tilde{G}}$ is a high-pass filter). We consider here for simplicity of subsequent notations the case in which G (the Fourier transform of $\tilde{G}$) is positive definite, rather than conditionally positive definite (Micchelli, 1986), and therefore is a bell-shaped function. It is possible to show (see the paper by Girosi, Jones and Poggio, 1995, for a sketch of the proof) that the function that minimizes the functional (2) is a classical Radial Basis Functions approximation scheme (Micchelli, 1986; Moody and Darken, 1989):

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i G(\mathbf{x} - \mathbf{x}_i) \tag{3}$$

where the vector of coefficients $(\mathbf{c})_i = c_i$ satisfies the following linear system:

$$(G + \lambda I)\mathbf{c} = \mathbf{y} \tag{4}$$

where I is the identity matrix, and we have defined the vector of output values $(\mathbf{y})_i = y_i$ and the matrix $(G)_{ij} = G(\mathbf{x}_i - \mathbf{x}_j)$. Classical examples of basis functions G include the Gaussian ($G(\mathbf{x}) = \exp(-\|\mathbf{x}\|^2)$) and the inverse multiquadric ($G(\mathbf{x}) = (1 + \|\mathbf{x}\|^2)^{-\frac{1}{2}}$). In the next section we will show how to embed the prior knowledge about radial symmetry in this framework and we will derive the corresponding solution.

4.2 Regularization Theory in Presence of Radial Symmetry

In the standard regularization theory approach, the minimization of the functional $H[f]$ is usually done on the space of functions Φ for which $\phi[f]$ is finite. If additional knowledge of the solution is known, that can be used to further constrain the space of solutions. If we know that the solution is a function with radial symmetry, then we can restrict ourselves to minimize $H[f]$ over $\Phi \bigcap \mathcal{R}$, where $\mathcal{R}$ is the set of radial functions. The problem we have to solve now is therefore the following:

$$\min_{f \in \Phi \bigcap \mathcal{R}} H[f] = \min_{f \in \Phi \bigcap \mathcal{R}} \sum_{i=1}^{N} (f(\mathbf{x}_i) - y_i)^2 + \lambda \phi[f] \ . \tag{5}$$

We now notice that any function in $\mathcal{R}$ uniquely defines a one dimensional function f^* as follows

$$f(\mathbf{x}) \equiv f^*(\|\mathbf{x}\|) \ . \tag{6}$$

Using this notation and standard results from Fourier theory, we can represent elements of $\mathcal{R}$ by their Hankel transform (Dautray and Lions, 1988)

$$f(\mathbf{x}) = C \|\mathbf{x}\|^{-\frac{d}{2}+1} \int ds \ \tilde{f}^*(s) s^{\frac{d}{2}} J_{\frac{d}{2}-1}(s\|\mathbf{x}\|) \tag{7}$$

where C is a known number, $J_{\frac{d}{2}-1}$ is a Bessel function of the first kind (Gradshtein and Ryzhik, 1981) and $\tilde{f}^*(s)$ is defined by $\tilde{f}(\mathbf{s}) \equiv \tilde{f}^*(\|\mathbf{s}\|)$. The functional of eq. (5) can now be thought as a functional of $\tilde{f}^*(s)$, and the solution of the minimization problem can be found by imposing the stationarity condition $\frac{\delta H[f]}{\delta \tilde{f}^*(s)} = 0$. After some lengthy calculations it is found that the solution of the approximation problem can be written in the following form:

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i H(\|\mathbf{x}\|, \|\mathbf{x}_i\|) \tag{8}$$

where we have defined

$$H(\|\mathbf{x}\|, \|\mathbf{x}_i\|) = (\|\mathbf{x}\| \|\mathbf{x}_i\|)^{-\frac{d}{2}+1} \int ds \tilde{G}^*(s) s J_{\frac{d}{2}-1}(s\|\mathbf{x}\|) J_{\frac{d}{2}-1}(s\|\mathbf{x}_i\|) \tag{9}$$

Although the basis function H does not have a friendly look, notice the similarity of the solution (8) with the standard solution (3). In both cases the final approximating function is a linear superposition of basis functions, and there is one basis function for each data point. From a computational point of view, in both cases the coefficients c_i are found by solving a linear system, with the only difference that in the case (8) the matrix $(G)_{ij}$ of eq. (4) is replaced by the matrix $(H)_{ij} = H(\|\mathbf{x}_i\|, \|\mathbf{x}_j\|)$. However, while it is clear that the standard solution is obtained by placing a "bump" function at each data point, this interpretation is not evident from the solution (8). As the following example shows, a very similar thing happens indeed, and this will become clearer in the next section, when we will discuss the creation of "virtual" examples.

Example Let us consider the very common case in which the basis function $G(\mathbf{x})$ is Gaussian. In this case its Fourier transform is also a Gaussian, and therefore $G^*(s) = \exp(-s^2)$. The integral of eq. (9) can be performed (Gradshtein and Ryzhik, 1981), to obtain the following form for H:

$$H(\|\mathbf{x}\|, \|\mathbf{x}_i\|) = e^{-(\|\mathbf{x}\|^2 + \|\mathbf{x}_i\|^2)} I_{\frac{d}{2}-1}(2\|\mathbf{x}\|\|\mathbf{x}_i\|) \tag{10}$$

where $I_{\frac{d}{2}-1}$ is the Bessel function of first kind of imaginary argument (Gradshtein and Ryzhik, 1981, par. 8.406). A plot of this function in 2 dimensions is presented in figure (2), where we have set $\|\mathbf{x}_i\| = 2$. It is clear that this function is a radial "bump" function, whose bump is concentrated on a circle of radius $\|\mathbf{x}_i\|$. Any radial section of this function looks like a Gaussian function centered at $\|\mathbf{x}_i\|$, providing a local, radially symmetric, form of approximation.

4.3 Radial Symmetry and "Virtual" Examples

In this section, we use the prior knowledge to generate new, "virtual" examples, from the existing data set.

Let $D = \{(\mathbf{x}_i, y_i) \in R^d \times R\}_{i=1}^N$ be our data set, and let us assume that we know that the function h underlying the data has radial symmetry. This means that $f(\mathbf{x}) = f(R_\theta \mathbf{x})$ for all the possible rotation matrices R_θ in d dimensions. Here θ is a $d-1$ dimensional vector of parameters that represents a point of Σ_{d-1}, the surface of the d-dimensional unit sphere. This property implies that if $(\mathbf{x}_i, y_i)$ is an example of h, the points $(R_\theta \mathbf{x}_i, y_i)$, for all $\theta \in \Sigma_{d-1}$, are also examples of h, and we call these additional points the "virtual" examples.

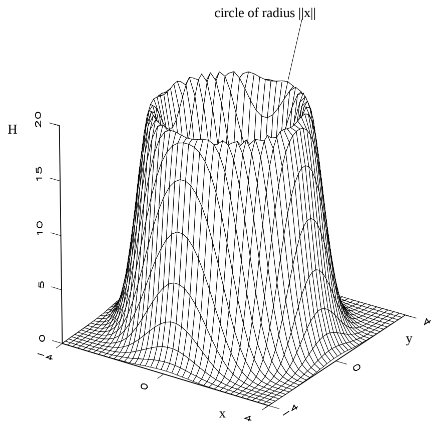

Figure 2: The basis function $H(\|\mathbf{x}\|, \|\mathbf{x}_i\|)$, for $\mathbf{x}_i = (2, 0)$.

Let us now consider a standard Radial Basis Functions approximation technique, of the form (3). Suppose for the moment that the function is invariant with respect to a *finite* number of rotations $R_{\theta_1}, \ldots, R_{\theta_n}$. Each example $\mathbf{x}_i$ will therefore generate n virtual examples $R_{\theta_1}\mathbf{x}_i, \ldots R_{\theta_n}\mathbf{x}_i$, that can now be included in the expansion (3) together with the regular examples. It is trivial to see that, because of the invariance property of h, the coefficients of the basis functions corresponding to the virtual examples will be equal to the coefficients of the corresponding, original example. As a result we have that eq. (3) has to be replaced by

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i \sum_{\alpha=0}^{n} G(\mathbf{x} - R_{\theta_\alpha}\mathbf{x}_i)$$

where we have defined $\theta_0 = 0$, so that $R_{\theta_0}\mathbf{x}_i = \mathbf{x}_i$. We now relax the assumption that the function is invariant with respect to only a finite number of rotations, and allow θ to span the entire surface Σ_{d-1}. The equation above suggests to replace eq. (3) with the following:

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i \int_{\Sigma_{d-1}} d\Omega_{d-1}(\theta)\ G(\mathbf{x} - R_\theta \mathbf{x}_i) \tag{11}$$

where $d\Omega_{d-1}(\theta)$ is the uniform measure over Σ_{d-1}. Using the Hankel representation (7) for the radial function G in eq. (11), the integral over Σ_{d-1} can be performed, and provides the result:

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i H(\|\mathbf{x}\|, \|\mathbf{x}_i\|)$$

where $H(\|\mathbf{x}\|, \|\mathbf{x}_i\|)$ is given precisely be expression (9)! From this derivation it is clear that the basis function $H(\|\mathbf{x}\|, \|\mathbf{x}_i\|)$ is an infinite superposition of Gaussian functions, whose centers uniformly cover the surface of the sphere of radius $\|\mathbf{x}_i\|$.

Therefore creating virtual examples seems to be, in a sense, the "right thing" to do, leading to the same result that one gets from the more "principled" and sophisticated approach of regularization theory. The appealing feature of the virtual examples technique is the fact that it can be applied in very general cases, in which it might be impossible to derive analytical results as the one derived in section 3.

5 Virtual Examples in Vision and Speech

The goal of this paper is to suggest the creation of virtual examples as a technique to incorporate prior information in machine learning problems. The previous section shows how creating virtual examples can be equivalent to incorporation of the prior information as a regularizer within a framework for function learning. Thus the virtual example strategy can often be more than a good heuristic. We now turn our attention to some real world problems that arise in computer vision and speech recognition and give examples of how one might generate virtual examples under certain conditions.

In the examples we are about to consider, the non-trivial part of the virtual example strategy is identifying the set of legal transformations that allow a new, valid, example to be created. In previous treatments of the machine learning problem that concentrated on function learning, the legal transformations were typically very simple and could easily be used to create examples. For example, whether the function is even or odd, or whether it has radial symmetry is easy to deal with. Imagine, instead, that one were interested in object recognition. How does one generate a new example? There are certain obvious cases. For example, by translating the image in the image plane or dilating the image (scale transformation) one could generate some trivial cases of new examples. However, there are some other non obvious ones like rotation in depth, or changing the expression of a face that are significantly harder to realize. In the next section, we discuss the problem of object recognition, how to view it within a function learning paradigm, and how to generate non-trivial virtual examples for it.

5.1 Virtual Views for Object Recognition

Consider the problem of recognizing 3D objects from their 2D images. A particular class of 3D objects (like cars, or cubes, or faces) can be defined in terms of pointwise features that have been put in correspondence with each other. If, for example, there are n features, and each feature is represented by its location in a 3D coordinate system (say by its x, y, z coordinates) then a particular view of a particular 3D object can be represented as a point in R^{3n}. However, note that not all points in R^{3n} correspond to valid views of 3D objects. Trying to learn this object class could be regarded as trying to learn a characteristic function in R^{3n}, i.e., a function of the form:

$$1_E(x) : R^{3n} \longrightarrow \{0,1\} = \begin{array}{ll} 1 & x \in E \subset R^{3n} \\ 0 & \text{otherwise} \end{array}$$

When the 3D view is projected to 2D, then each 2D view can now be represented as a characteristic function over R^{2n}. For problems such as these, one can rarely specify simple mathematical constraints (like radial symmetry etc.) on the characteristic functions. This makes the recognition problem particularly challenging. Consider, for example, the face recognition problem studied by Beymer (1994). The goal is to recognize faces of different people under a variety of views. One approach to this is to collect a large number of views from each person and train a classifier to recognize them. Shown in fig. 3 are fifteen views of one particular face that have been collected as training examples for that face. This relatively straightforward approach works but usually requires a large number of training examples.

In contrast, an alternative strategy is to use some kind of prior knowledge about the class of faces in order to generate virtual examples or virtual "views". One could then train a view independent system on the basis of these virtual examples. This raises the question: if we are given examples of images belonging to some class, then can we generate new examples of images belonging to the same class? In order to do this we need to uncover the set of *legal* transformations that allow us to take elements of E and come up with other elements of E. Prior knowledge about the class of objects allow us to uncover such a set of valid transformation.

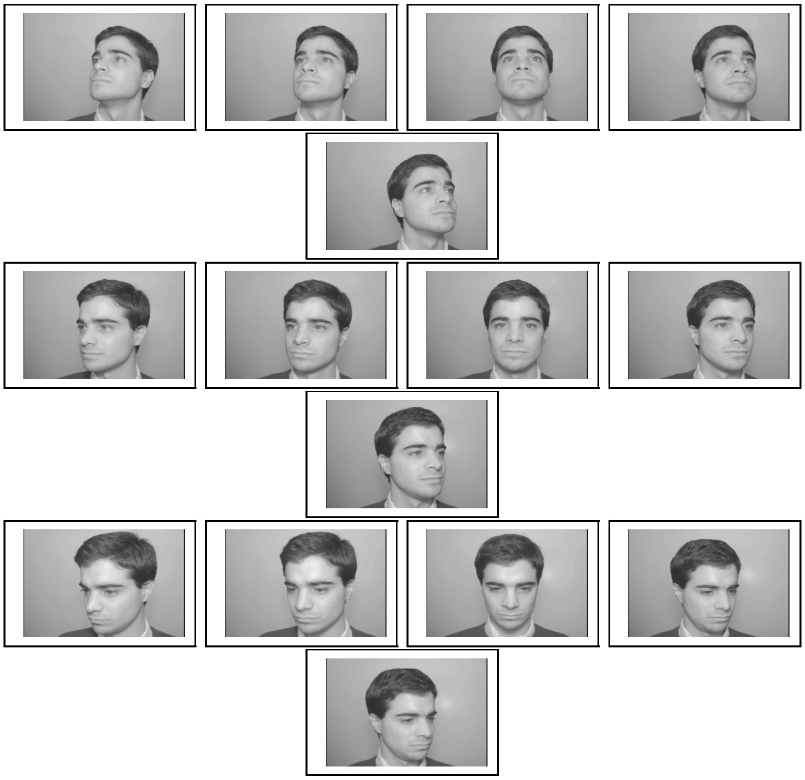

Figure 3: The pose-invariant, view-based face recognizer uses 15 views to model a person's face. From Beymer, 1994.

5.2 Symmetry as Prior Information

Poggio and Vetter (1992) examined in particular the case of bilateral symmetry of certain 3D objects, such as faces. Suppose that we have a model 2D view of an object and a pair of symmetric points in this 2D view. For our purposes, we can define an object to be *bilaterally symmetric* if the following transformation of any 2D view of a pair of symmetric points of the object yields a *legal view* of the pair, that is the orthographic projection of a rigid rotation of the object

$$D\mathbf{x}_{pair} = \mathbf{x}^*_{pair} \tag{12}$$

with

$$\mathbf{x}_{pair} = \begin{pmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{pmatrix} \qquad \mathbf{x}^*_{pair} = \begin{pmatrix} -x_2 \\ -x_1 \\ y_2 \\ y_1 \end{pmatrix}$$

and

$$D = \begin{pmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Notice that symmetric pairs are the elementary features in this situations and points lying on the symmetry plane are degenerate cases of symmetric pairs.

Geometrically, this simply means that for bilaterally symmetric objects simple transformations of a 2D view yield other views that are *legal*. The transformations are similar to mirroring one view around an axis in the image plane, as shown in Figure 4 top (where the left image is “mirrored” into the right one) and correspond – but only for a bilaterally symmetric object – to proper rotations of a rigid 3D object and their orthographic projection on the image plane. Using the transformation of equation 12 an additional view is generated from the one model view. If the two views are linearly independent, then one can resort to the 1.5 views theorem[3]

[3] Using the notation introduced earlier, the set E defines the space of valid image views of a particular object. The 1.5 views theorem states essentially that E can be regarded as a 6-dimensional vector space. Furthermore this basis can be computed from two linearly independent views. For further details on this, see Poggio and Vetter (1992).

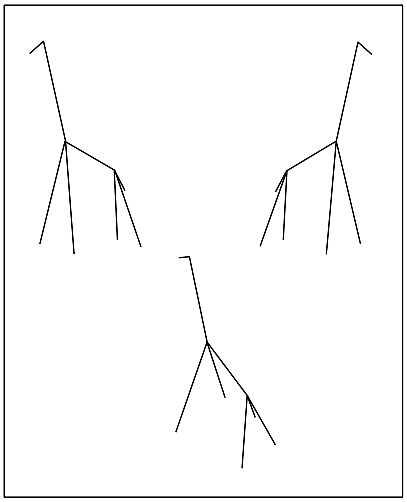

Figure 4: Given a single 2D view (upper left), a new view (upper right) is generated under the assumption of bilateral symmetry. The two views are sufficient to verify that a novel view (second row) corresponds to the same object as the first.

to compute a 3D basis that spans the space of the object. This allows us to compute a recognition function with just one true view. Bilateral symmetry has been used in face recognition systems (Beymer and Poggio, 1995) and psychophysical evidence supports its use by the human visual system (Schyns and Bulthoff, 1993; Troje and Bulthoff, 1995; Vetter, Poggio, and Bulthoff, 1994).

5.3 More General Transformations: Linear Object Classes

A more flexible way to acquire information about how images of objects of a certain class change under pose, illumination and other transformations, is to learn the possible pattern of variabilities and class-specific deformations from a representative training set of views of generic or prototypical objects of the same class – such as other faces. In particular, if the objects belong to a well behaved class known as a linear object class, the transformations can be easily learned. In this manner prior knowledge that the object class is linear can be utilized effectively to generate novel views that can be incorporated in the training process.

Although this approach of linear classes originates from the proposal of Poggio and Vetter (1992) for countering the curse-of-dimensionality in applications of supervised learning techniques, more powerful versions have been developed recently. Techniques based on non-linear learning networks have been developed by Beymer, Shashua and Poggio (1993) as well as Beymer and Poggio (1995). For our purposes here, we now provide a brief overview of the technique of linear classes for generating novel views of objects.

5.3.1 3D Objects, 2D Projections, and Linear Classes

Consider a 3D view of a three-dimensional object that is defined in terms of pointwise features (Poggio and Vetter, 1992). Such a 3D view can be represented by a vector $\mathbf{X} = (x_1, y_1, z_1, x_2, ..., y_n, z_n)^T$, that is by the x, y, z-coordinates of its n feature points. Further, assume that $\mathbf{X} \in \Re^{3n}$ is the linear combination of q 3D views $\mathbf{X}_i$ of *other* objects of the same dimensionality, such that:

$$\mathbf{X} = \sum_{i=1}^{q} \alpha_i \mathbf{X}_i. \tag{13}$$

Consider now some linear operator L associated with a desired uniform transformation such as for instance a specific rotation in 3D. Let us define $\mathbf{X}^r = L\mathbf{X}$ to be the rotated 3D view of object $\mathbf{X}$. Because of the linearity of the group of uniform linear transformations $\mathcal{L}$, it follows that

$$\mathbf{X}^r = \sum_{i=1}^{q} \alpha_i \mathbf{X}_i^r \tag{14}$$

Thus, *if a 3D view of an object can be represented as the weighted sum of views of other objects, its rotated view is a linear combination of the rotated views of the other objects with the same weights.* Of course for an arbitrary 2D view that is a projection of a 3D view, a decomposition like (13) does not in general imply a decomposition of the rotated 2D views (it is a necessary but not a sufficient condition).

Linear Classes
A natural question to ask, therefore, is: "Under what conditions do the 2D projections of 3D objects satisfy equation (13) to (14)?" The answer will clearly depend on the types of objects we use and also on the projections we allow. In a series of articles (Poggio and Vetter, 1992; Vetter and Poggio, 1995) the notion of linear classes has been introduced and developed — we provide a definition below:

A set of 3D views (of objects) $\{\mathbf{X}_i\}$ *is a* **linear object class** *under a linear projection P if* $dim\{\mathbf{X}_i\} = dim\{\mathbf{P}\mathbf{X}_i\}$ *with* $\mathbf{X}_i \in \Re^{3n}$ *and* $\mathbf{P}\mathbf{X}_i \in \Re^p$

and $p < 3n$

This is equivalent to saying that the minimal number of basis objects necessary to represent an object is not allowed to change under the projection. Note that the linear projection P is not restricted to projections from $3D$ to $2D$, but may also "drop" occluded points. Now assume $\mathbf{x} = P\mathbf{X}$ and $\mathbf{x}_i = P\mathbf{X}_i$ are the projections of elements of a linear object class with

$$\mathbf{x} = \sum_{i=1}^{q} \alpha_i \mathbf{x}_i \tag{15}$$

then $\mathbf{x}^r = P\mathbf{X}^r$ can be constructed without knowing $\mathbf{X}^r$using α_i of equation (15) and the given $\mathbf{x}_i^r = P\mathbf{X}_i^r$ of the other objects.

$$\mathbf{x}^r = \sum_{i=1}^{q} \alpha_i \mathbf{x}_i^r. \tag{16}$$

5.3.2 Implications

The relations described earlier suggest that we can use "prototypical" 2D views (the projections of a basis of a linear object class) and their known transformations to synthesize an operator that will transform a 2D view into a new 2D view when the object is a linear combination of the prototypes. In other words we can compute a new 2D view of such an object without knowing explicitly its three-dimensional structure. Notice also, that knowledge of the correspondence between equation (15) and equation (16) is not necessary (rows in a linear equation system can be exchanged freely). Therefore, the technique does not require one to compute the correspondence between views from different viewpoints. In fact some points may be occluded. Fig. 5 shows a very simple example of a linear object class and the construction of a new view of an object. Since the dimension of the class of all cuboids is 3, any cuboid can be represented as a linear combination of 3 prototypical cuboids. Thus the class is linear under all orthographic projections that preserve the three dimensions.
Remark: Three-dimensional objects differ in shape as well as in texture. To truly apply the linear class idea to gray level images, we need to derive object representations that incorporate texture. This can be done by developing shape and texture vector representations that are in correspondence and using the linear class idea over both.

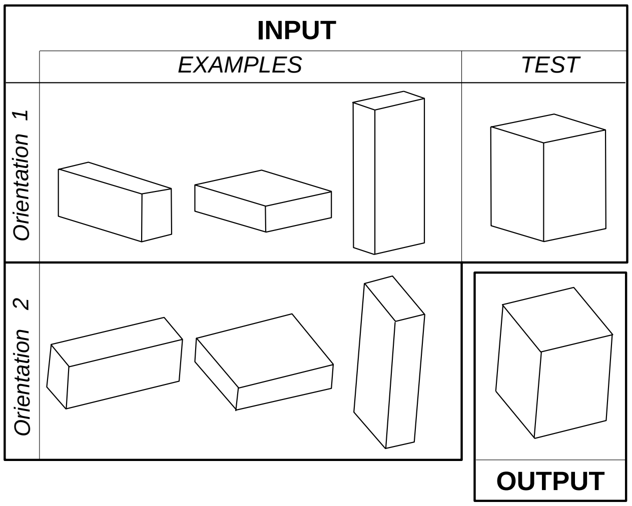

Figure 5: *Learning an image transformation according to a rotation of three-dimensional cuboids from one orientation (upper row) to a new orientation (lower row). The 'test' cuboid (upper row right) can be represented as a linear combination of the two-dimensional coordinates of the three example cuboids in the upper row. The linear combination of the three example views in the lower row, using the coefficients evaluated in the upper row, results in the correct transformed view of the test cuboid as output (lower row right). Notice that correspondence between views in the two different orientations is not needed and different points of the object may be occluded in the different orientations.*

5.3.3 Learning the Transformation

We finally complete the story by discussing how the transformation from a reference view to a novel view can be learned. Before we proceed, let us introduce a helpful change of coordinate systems in equations (15) and (16). Instead of using an absolute coordinate system, we represent the views as the difference to the view of a reference object of the same class. Subtracting the projection of a reference object from both sides of equations (15) and (16), we have

$$\Delta \mathbf{x} = \sum_{i=1}^{q} \alpha_i \Delta \mathbf{x}_i \tag{17}$$

and

$$\Delta \mathbf{x}^r = \sum_{i=1}^{q} \alpha_i \Delta \mathbf{x}_i^r. \tag{18}$$

After this change in the coordinate system, equation (18) now evaluates to the new difference vector to the rotated reference view. The new view of the object can be constructed by adding this difference to the reference view.

Steps in Constructing a Novel View:

Step 1: First, we compute the coefficients α_i for the optimal decomposition (in the sense of least square). We decompose an "initial" field $\Delta \mathbf{x}$ to a new object X into the "initial" fields $\Delta \mathbf{x}_i$ to the q given prototypes by minimizing

$$||\Delta \mathbf{x} - \sum_{i=1}^{q} \alpha_i \Delta \mathbf{x}_i||^2. \tag{19}$$

Rewriting this as $\Delta \mathbf{x} = \mathbf{\Phi} \boldsymbol{\alpha}$ (where $\mathbf{\Phi}$ is the matrix formed by the q vectors $\Delta \mathbf{x}_i$ arranged column-wise and $\boldsymbol{\alpha}$ is the column vector of the α_i coefficients) and minimizing equation (19) gives

$$\boldsymbol{\alpha} = (\mathbf{\Phi})^{+} \Delta \mathbf{x}. \tag{20}$$

Step 2: The observation of the previous section implies that the operator L that transforms $\Delta \mathbf{x}$ into $\Delta \mathbf{x}^r$ through $\Delta \mathbf{x}^r = L \Delta \mathbf{x}$, is given by

$$\Delta \mathbf{x}^r = \mathbf{\Phi}^r \boldsymbol{\alpha} = \mathbf{\Phi}^r \mathbf{\Phi}^{+} \Delta \mathbf{x} \qquad as \qquad L = \mathbf{\Phi}^r \mathbf{\Phi}^{+} \tag{21}$$

and thus can be learned from the 2D example pairs $(\Delta \mathbf{x}_i, \Delta \mathbf{x}_i^r)$. In this case, a one-layer, linear network (compare Hurlbert and Poggio, 1988) can be used to learn the transformation L. L can then transform a view of a novel object of the same class. If the q examples are linearly independent $\mathbf{\Phi}^{+}$ is given by $\mathbf{\Phi}^{+} = (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T$; in the other cases equation (19) can be solved by an SVD algorithm.

Step 3: Analogous steps have to be taken to deal with textures. Before decomposing the new texture into example textures, all textures are mapped onto a common basis — typically, the reference image using correspondences. In this representation the decomposition of the textures can be performed as described above.

Step 4: The final step is image rendering. The α coefficients that are computed for both texture and shape vectors are then applied to the prototype examples in the second orientation. The correspondence fields to the new image are combined with the reference image (often using forward warping; Wolberg, 1990) to generate the novel image.

Using this procedure, we can now generate novel views of images with prior knowledge the image belongs to a linear class. Fig. 5 shows a case where a new view has been generated for the class of cuboids for which the linear class assumption is correct. More interestingly, however, fig. 6 shows how novel views of a face can be generated from prototypical views using the linear class technique. While faces are not theoretically guaranteed to constitute a linear class, in practice, the assumption turns out to be quite good as the example shows. Thus, instead of collecting fifteen example patterns and training on them, one could generate virtual examples using the techniques described here and train on a combination of the real and virtual examples. A system based on this has been successfully implemented and described in Beymer and Poggio, 1995. A face recognition system using a single real view plus 14 virtual views as in figure 6 (per person) achieved a recognition rate of 85% correct while a system using 15 real views (per person) achieved 99% on the same database. Both systems were significantly better than a system using one real view per person (32%). Notice that in the approach outlined above, correspondence plays a key role. Interestingly, correspondence for the prototypes is only required between views from the same viewpoint. Correspondence is also required between the real image and one of the prototypes and can be computed automatically by optical flow techniques (Beymer and Poggio, 1996). A different approach (see Jones and Poggio, 1995) does not require an explicit correspondence between the real image and the prototypes. Vetter et al (1997) propose a technique that may also allow for an automatic correspondence between the prototype images.

5.4 Virtual Examples in Speech Recognition

Another example of the potential utility of the "virtual example" technique for incorporating prior information can be provided in the context of speech recognition. In the production of speech by humans, an important source of prior information lies in the physical constraints that the vocal tract and speech articulators (speech producing apparatus) necessarily have to obey. For example, different sounds (phonemes) produced by the same speaker might share some common characteristics related to the properties of the

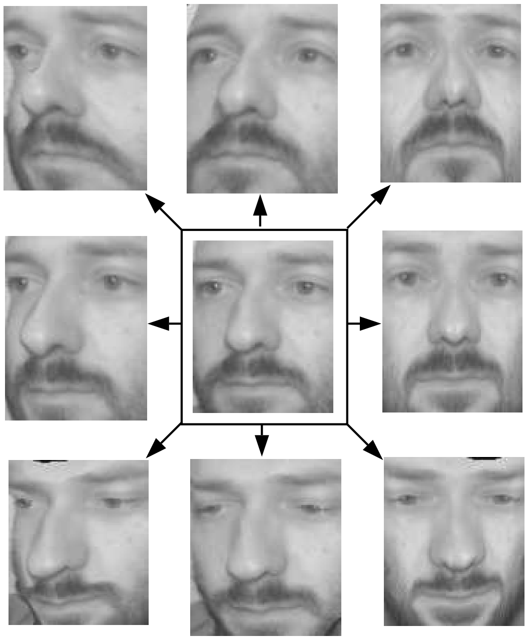

Figure 6: A real view (center) surrounded by virtual views derived from it a technique related to the linear class technique but even simpler. The correspondence between the real view and the prototypes is computed analytically. For details on the required process, see Beymer and Poggio, 1995.

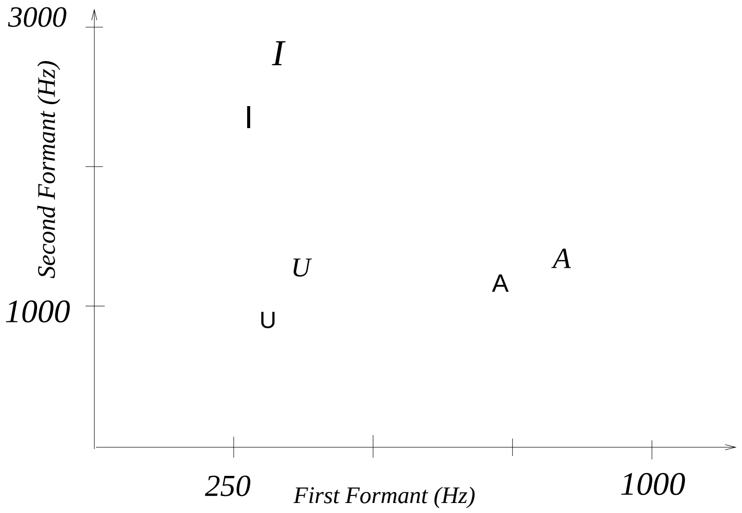

Figure 7: The first and second formants of "i" (as in *beat*), "a" (as in *palm*) and "u" (as in *boot*). The values for the female speaker are plotted in italics.

individual speaker's vocal tract. Thus, if a speaker has a high pitch, then this is likely to be the case for all the phonemes the speaker produces.

Consider fig. 7 which compares the formant values for three vowel sounds — "i" (as in *beat*, "a" (as in *palm*) and "u" (as in *boot*) for one male and one female speaker. The data has been obtained from Peterson and Barney (1952). In speech recognition, one will have to distinguish between the different vowels on the basis of certain vowel features like formants. Notice that while the broad pattern of formants for the two speakers are the same, the actual formant values differ — the female speaker has consistently higher formants in general. It turns out that the formant values are related to the length of the vocal tract — people with longer tracts have lower formants and vice-versa. This is the sort of prior information that one would like to capture while attempting to solve the speech recognition problem in a way that is invariant to the systematic speaker differences.

Roughly speaking, speech is produced when air is pumped into the vocal tract thereby exciting it and the corresponding acoustic waves are transmitted through the air to the hearer. Thus, the vocal tract (including the nasal tract) modulates the excitations produced from below resulting in speech. From an acoustic standpoint, the vocal tract has been modeled as a non-uniform tube and its resonances correspond to the formants described earlier. The length of the tube is inversely related to the frequency of the resonances and thus humans with longer vocal tracts have formants at lower frequencies. Each different sound corresponds to a different articulatory configuration, a correspondingly different vocal tract shape, a non-uniform tube of different shape, and therefore a correspondingly different set of formant values. All speakers producing an "a" would have roughly the same configuration — as a first order approximation, speaker differences would come about due to variations in scale, i.e., all the vocal tracts would have the same overall shape, the vocal tract of a large man would be longer than that of a child. This overall length, for example, is a quantity that is preserved across all sounds of the speaker. If it could be extracted automatically from a speaker's "a", then it could be used to scale a canonical speaker's "i" to produce a novel "i".

The acoustic problem can be modeled using electrical circuits and the most common formulation has taken a source-filter point of view. The vocal tract is viewed as a filter shaping the input provided by electrical sources. Sources are roughly of two types: (1) *periodic* sources that correspond to the glottal vibrations during voiced speech, and (2) *aperiodic* sources that correspond to various kinds of turbulent sources produced during unvoiced (or partially voiced) speech, e.g., during fricatives like "s" etc. Shown in fig. 8 is a schematic view of the speech production apparatus. The vocal tract filter $H(z)$ is shown to be parameterized by two kinds of parameters, p, that models the shape of the tract and depends upon the phonetic identity of the sound, and s, that models things like overall scale and depends upon the specific characteristics of the speaker. Similarly, the voiced periodic source is also parameterized by a set of parameters (denoted by w) that are speaker-specific and don't change from phoneme to phoneme. Typical examples of such speaker-specific parameters are pitch, voice quality, etc.

One approach to the virtual examples idea is to obtain a number of examples of a novel speaker's "a" sounds. For such sounds, we already know the phoneme-dependent parameter values p (since this depends only on the phoneme and is common to all speakers, such parameters can be estimated directly from the data collected from a reference speaker). From the novel speaker's sounds, we estimate the speaker-specific parameters, s. Now, to generate new instances of a speaker's "i", we can drive the speech production model using the phoneme-specific parameters (p that are derived from the reference speaker) and the speaker-specific parameters (s) that are

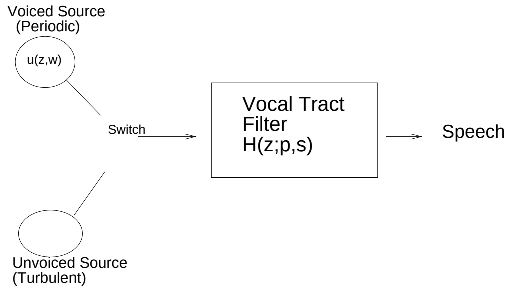

Figure 8: A schematic view of the speech production apparatus.

derived from the test speaker. Note that such a virtual example strategy would depend in large part upon the fidelity of the speech production model that one has in the first place.

A simpler and more direct strategy is to learn the mapping from "a" to "i" by collecting examples of each from a number of different speakers. By doing this, we can convert a speaker's "a" into a novel speaker's "i" using the functional mapping learned. Examine fig. 9 that shows the relationship between "ae" (as in *bat*) and "i" (as in *beat*) from the same speaker. Data from 360 speakers were collected and speakers were grouped into 9 speaker classes. A weighted spectral measure was computed for each of the speaker's "ae" and "i" sounds. The mean values of this measure for each of the speaker groups is plotted. Notice the strong correlation between the mean value of a group's "ae" and the mean value of that group's "i" suggesting strong predictability. Such an idea has been used successfully in incorporating speaker information into a speech recognition system (Niyogi and Zue, 1991; Niyogi, 1992).

The problem described here is analogous to the vision example we described. Different poses of the same face are like different sounds of the same speaker (equivalent to different "poses" of the vocal tract). In object recognition, we used knowledge of the relationship between different poses of prototypes to create a novel pose for a new speaker. In speech recognition, we used knowledge of the relationship between different sounds of prototypical speakers to create a novel sound for a new speaker. This particular strategy of creating virtual examples has rarely been used in speech recognition — and would be particularly useful if one wishes to adapt a speech

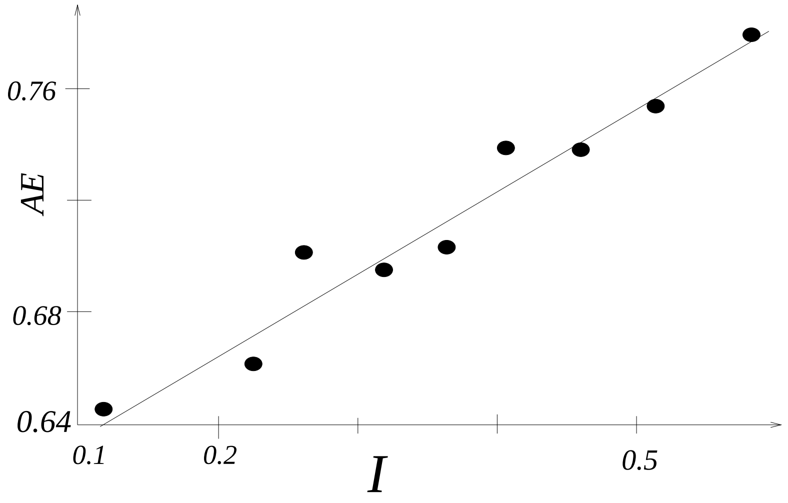

Figure 9: A weighted spectral average measure for a speaker's "i" and the same speaker's "ae" plotted against each other. Data from 360 speakers were collected and grouped into 9 classes and the class average for each of the two sounds has been plotted. Notice the strong correlation suggesting that it is possible to predict the "ae" sound from the "i" sound given the group to which a speaker belongs.

recognition system to a new speaker using extremely limited amounts of adaptation data.

6 Conclusions

In this paper, we introduced the idea of virtual examples as a possible strategy for incorporating prior knowledge about the target function in a learning from examples paradigm. We motivated the importance of using virtual examples by discussing the issue of sample complexity in machine learning. Specifically, using the general results of Vapnik and Chervonenkis, we argued that the number of examples needed for successful generalization would be prohibitive without adequate constraints on the hypothesis class. However, successful learning would result only if the such constraints properly reflected our prior knowledge of the problem being solved.

The creation of virtual examples is one way around the dilemma. We showed how the idea of virtual examples was mathematically equivalent to incorporating prior knowledge as a regularizer in function learning in certain restricted domains. In most substantive real world problems, however, it is rare that prior knowledge can be directly implemented as an elegant regularization constraint. For such cases, the creation of virtual examples might be more straightforward. However, this is not to say that virtual examples are easily created. In particular, in this paper, we spent a significant portion of time discussing the details of the creation of virtual examples for object recognition and speech recognition. The results obtained so far suggest that this might be a promising way to incorporate prior knowledge into an example-based learning framework leading to systems that generalize well from the limited amounts of data that is typically available in real world problems.

References

[1] Y. Abu-Mostafa. A method for learning from hints. In S. J. Hanson, Jack D. Cowan, and C. Lee Giles, editors, *Advances in Neural information processings systems 5*, San Mateo, CA, 1992. Morgan Kaufmann Publishers.

[2] Y.S. Abu-Mostafa. Learning from hints in neural networks. *Journal of Complexity*, 6:192–198, 1990.

[3] Y.S. Abu-Mostafa. Hints and the VC-dimension. *Neural Computation*, 5:278–288, 1993.

[4] Y.S. Abu-Mostafa. Hints. *Neural Computation*, 7:639–671, 1995.

[5] A.R. Barron. Approximation and estimation bounds for artificial neural networks. *Machine Learning*, 14:115–133, 1994.

[6] M. Bertero. Regularization methods for linear inverse problems. In C. G. Talenti, editor, *Inverse Problems*. Springer-Verlag, Berlin, 1986.

[7] D. Beymer and T. Poggio. Face recognition from one example view. In *Proceedings of the International Conference of Computer Vision*, Cambridge, MA, June 1995.

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

[9] D. Beymer, A. Shashua, and T. Poggio. Example based image analysis and synthesis. A.I. Memo No. 1431, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1993.

[10] David J. Beymer. Face recognition under varying pose. In *Proceedings of IEEE Conf. on Computer Vision and Pattern Recognition*, Seattle, WA, 1994.

[11] C. M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural Computation*, 7:108–116, 1995.

[12] L. Breiman, J. H. Friedman, R.A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, CA, 1984.

[13] R. Dautray and J.L. Lions. *Mathematical Analysis and Numerical Methods for Science and Technology*, volume 2. Springer-Verlag, Berlin, 1988.

[14] D. Wolpert (ed.). *The Mathematics of Generalization.* Addison-Wesley, Reading, MA, 1995.

[15] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7:219–269, 1995.

[16] I.S. Gradshteyn and I.M. Ryzhik. *Table of integrals, series, and products.* Academic Press, New York, 1981.

[17] A. Hurlbert and T. Poggio. Synthesizing a color algorithm from examples. *Science*, 239:482–485, 1988.

[18] M. Jones and T. Poggio. Model-based matching of line drawings by linear combination of prototypes. In *Proceedings of the International Conference on Computer Vision*, pages 531–536. IEEE, June 1995.

[19] T. K. Leen. From data distributions to regularization in invariant learning. *Neural Computation*, 7:974, 1995.

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

[21] J. Moody and C. Darken. Learning with localized receptive fields. In G. Hinton, T. Sejnowski, and D. Touretzsky, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 133–143, Palo Alto, 1988.

[22] J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.

[23] V.A. Morozov. *Methods for solving incorrectly posed problems.* Springer-Verlag, Berlin, 1984.

[24] Joseph L. Mundy and Andrew Zisserman. *Geometric Invariance in Computer Vision.* MIT Press, Cambridge, MA, 1992.

[25] P. Niyogi. Modelling speaker variability and imposing speaker constraints in phonetic classification. Technical Report TR-533, MIT Laboratory for Computer Science, 1992.

[26] P. Niyogi and F. Girosi. On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions. *Neural Computation*, 8:819–842, 1996.

[27] P. Niyogi and V.W. Zue. Correlation analysis of vowels and its application to speech recognition. In *Proceedings of Eurospeech*, Genoa, Italy, 1991.

[28] G. E. Peterson and H. L. Barney. Control methods used in a study of the vowels. *Journal of the Acoustical Society of America*, 24(2):175–184, 1952.

[29] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9), September 1990.

[30] T. Poggio and T. Vetter. Recognition and structure from one 2D model view: observations on prototypes, object classes and symmetries. A.I. Memo No. 1347, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1992.

[31] D. Pomerleau. Alvinn: an autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems, I*, San Mateo, CA, 1989. Morgan Kaufman.

[32] D. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Parallel Distributed Processing.* MIT Press, Cambridge, MA, 1986.

[34] P.G. Schyns and H. Bülthoff. Conditions for viewpoint dependent face recognition. A.I. Memo 1432, MIT Artificial Intelligence Lab., Cambridge, MA, August 1993.

[35] P. Simard, Y. LeCun, and J. Denker. Efficient pattern recognition using a new transformation distance. In *Advances in Neural Information Processing Systems V*, pages 50–58, San Mateo, CA, 1993. Morgan Kaufmann Publishers.

[36] P. Simard, B. Victorri, Y. LeCun, and J. Denker. Tangent-Prop - A formalism for specifying selected invariances in an adaptive network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems IV*, pages 895–903, San Mateo, CA, 1992. Morgan Kaufmann Publishers.

[37] A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math. Dokl.*, 4:1035–1038, 1963.

[38] N. Troje and H.H. Bülthoff. Face recognition under varying pose: The role of texture and shape. *submitted*, 1995.

[39] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, New York, 1995.

[40] V. N. Vapnik. *Estimation of Dependences Based on Empirical Data.* Springer-Verlag, Berlin, 1982.

[41] A. Verri and T. Poggio. Regularization theory and shape constraints. A.I. Memo No. 916, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1986.

[42] T. Vetter, M. Jones, and T. Poggio. A bootstrapping algorithm for learning linear models of object classes. In *Proceedings of the 1997 Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 40–46, June 1997.

[43] T. Vetter, T. Poggio, and H.H. Bülthoff. The importance of symmetry and virtual views in three-dimensional object recognition. *Current Biology*, 4:18–23, 1994.

[44] G. Wahba. *Splines Models for Observational Data.* Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.

[45] Georg Wolberg. *Image Warping.* IEEE Computer Society Press, Los Alamitos CA, 1990.

æ