January 3, 2003

Draft for the Notices of the AMS

The Mathematics of Learning: Dealing with Data

Tomaso Poggio[†] **and Steve Smale**[‡]

CBCL, McGovern Institute, Artificial Intelligence Lab, BCS, MIT[†]

Toyota Technological Institute at Chicago and Professor in the Graduate School,
University of California, Berkeley[‡]

Abstract

Learning is key to developing systems tailored to a broad range of data analysis and information extraction tasks. We outline the mathematical foundations of learning theory and describe a key algorithm of it.

1 Introduction

The problem of understanding intelligence is said to be the greatest problem in science today and "the" problem for this century - as deciphering the genetic code was for the second half of the last one. Arguably, the problem of learning represents a gateway to understanding intelligence in brains and machines, to discovering how the human brain works and to making intelligent machines that learn from experience and improve their competences as children do. In engineering, learning techniques would make it possible to develop software that can be quickly customized to deal with the increasing amount of information and the flood of data around us.

Examples abound. During the last decades, experiments in particle physics have produced a very large amount of data. Genome sequencing is doing the same in biology. The Internet is a vast repository of disparate information which changes rapidly and grows at an exponential rate: it is now significantly more than 100 Terabytes, while the Library of Congress is about 20 Terabytes.

We believe that a set of techniques, based on a new area of science and engineering becoming known as "supervised learning" – will become a key technology to extract information from the ocean of bits around us and make sense of it.

Supervised learning, or learning-from-examples, refers to systems that are trained, instead of programmed, with a set of examples, that is a set of input-output pairs. Systems that could learn from example to perform a specific task would have many applications. A bank may use a program to screen loan applications and approve the "good" ones. Such a system would be trained with a set of data from previous loan applications and the experience with their defaults. In this example, a loan application is a point in a multidimensional space of variables characterizing its properties; its associated output is a binary "good" or "bad" label.

In another example, a car manufacturer may want to have in its models, a system to detect pedestrians that may be about to cross the road to alert the driver of a possible danger while driving in downtown traffic. Such a system could be trained with positive and negative examples: images of pedestrians and images without people. In fact, software trained in this way with thousands of images has been recently tested in an experimental car of Daimler. It runs on a PC in the trunk and looks at the road in front of the car through a digital camera [1].

Algorithms have been developed that can produce a diagnosis of the type of cancer from a set of measurements of the expression level of many thousands human genes in a biopsy of the tumor measured with a cDNA microarray containing probes for a number of genes. Again, the software learns the classification rule from a set of examples, that is from examples of expression patterns in a number of patients with known diagnoses. The challenge, in this case, is the high dimensionality of the input space – in the order of $20,000$ genes – and the small number of examples available for training – around 50. In the future, similar learning techniques may be capable of some learning of a language and, in particular, to translate information from one language to another.

What we assume in the above examples is a machine that is trained, instead of programmed, to perform a task, given data of the form $(x_i, y_i)_{i=1}^m$. Training means synthesizing a function that best represents the relation between the inputs x_i and the corresponding outputs y_i. The central question of learning theory is how well this function generalizes, that is how well it estimates the outputs for previously unseen inputs.

As we will see later more formally, learning techniques are similar to fitting a multivariate function to a certain number of measurement data. The key point, as we just mentioned, is that the fitting should be *predictive*, in the same way that fitting experimental data (see figure 1) from an experiment in physics can in principle uncover the underlying physical law, which is then used in a predictive way. In this sense, learning is also a principled method for distilling predictive and therefore scientific "theories" from the data.

We begin by presenting a simple "regularization" algorithm which is important in learning theory and its applications. We then outline briefly some of its applications and its performance. Next we provide a compact derivation of it. We then provide general theoretical foundations of learning theory. In particular, we outline the key ideas of decomposing the generalization error of a solution of the learning problem into a sample and an approximation error

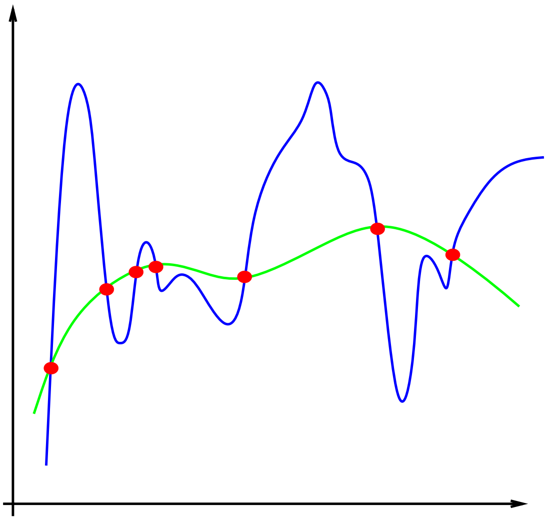

Figure 1: *How can we learn a function which is capable of generalization – among the many functions which fit the examples equally well (here $m = 7$)?*

component. Thus both probability theory and approximation theory play key roles in learning theory. We apply the two theoretical bounds to the algorithm and describe for it the tradeoff – which is key in learning theory and its applications – between *number of examples* and *complexity of the hypothesis space*. We conclude with several remarks, both with an eye to history and to open problems for the future.

2 A key algorithm

2.1 The algorithm

How can we fit the "training" set of data $S_m = (x_i, y_i)_{i=1}^m$ with a function $f : X \to Y$ – with X a closed subset of $\mathbb{R}^n$ and $Y \subset \mathbb{R}$ – that generalizes, eg is predictive? Here is an algorithm which does just that and which is almost magical for its simplicity and effectiveness:

1. Start with data $(x_i, y_i)_{i=1}^m$
2. Choose a symmetric, positive definite function $K_x(x') = K(x, x')$, continuous on $X \times X$. A kernel $K(t, s)$ is *positive definite* if $\sum_{i,j=1}^n c_i c_j K(t_i, t_j) \geq 0$ for any $n \in \mathbb{N}$ and choice of $t_1, ..., t_n \in X$ and $c_1, ..., c_n \in \mathbb{R}$. An example of such a Mercer kernel is the Gaussian

$$K(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}}. \tag{1}$$

restricted to $X \times X$.

3. Set $f : X \to Y$ to

$$f(x) = \sum_{i=1}^{m} c_i K_{x_i}(x). \tag{2}$$

where $\mathbf{c} = (c_1, ..., c_m)$ and

$$(m\gamma\mathbf{I} + \mathbf{K})\mathbf{c} = \mathbf{y} \tag{3}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{K}$ is the square positive definite matrix with elements $K_{i,j} = K(x_i, x_j)$ and $\mathbf{y}$ is the vector with coordinates y_i. The parameter γ is a positive, real number.

The linear system of equations 3 in m variables is well-posed since $\mathbf{K}$ is positive and $(m\gamma\mathbf{I} + \mathbf{K})$ is strictly positive. The condition number is good if $m\gamma$ is large. This type of equations has been studied since Gauss and the algorithms for solving it efficiently represent one the most developed areas in numerical and computational analysis.

What does Equation 2 say? In the case of Gaussian kernel, the equation approximates the unknown function by a weighted superposition of Gaussian "blobs" , each centered at the location x_i of one of the m examples. The weight c_i of each Gaussian is such to minimize a regularized empirical error, that is the error on the training set. The σ of the Gaussian (together with γ, see later) controls the degree of smoothing, of noise tolerance and of generalization. Notice that for Gaussians with $\sigma \to 0$ the representation of Equation 2 effectively becomes a "look-up" table that cannot generalize (it provides the correct $y = y_i$ only when $x = x_i$ and otherwise outputs 0).

2.2 Performance and examples

The algorithm performs well in a number of applications involving regression as well as binary classification. In the latter case the y_i of the training set $(x_i, y_i)_{i=1}^m$ take the values $\{-1, +1\}$; the predicted label is then $\{-1, +1\}$, depending on the sign of the function f of Equation 2.

Regression applications are the oldest. Typically they involved fitting data in a small number of dimensions [1]. More recently, they also included typical learning applications, sometimes with a very high dimensionality. One example is the use of an algorithm in computer graphics for synthesizing new images [1]. The inverse problem of estimating facial expression and object pose from an image is another successful application [1]. Still another case is the control of mechanical arms. There are also applications in finance, as, for instance, the estimation of the price of derivative securities, such as stock options. In this case, the algorithm replaces the classical Black-Scholes equation (derived from first principles) by learning the map from an input space (volatility, underlying stock price, time to expiration of the option etc.) to the output space (the price of the option) from historical data [1].

Binary classification applications abound. The algorithm was used to perform binary classification on a number of toy problems [1]. It was also used to perform visual object recognition in a view-independent way and in particular face recognition and sex categorization from face images [1]. Other applications span bioinformatics for classification of human cancer from microarray data, text summarization, sound classification[1].

Surprisingly, it has been realized quite recently that the same linear algorithm not only works well but is fully comparable in binary classification problems to the most popular classifiers of today (that turn out to be of the same family, see later).

2.3 Derivation

The algorithm described can be derived from Tikhonov regularization. To find the minimizer of the the error we may try to solve the problem – called Empirical Risk Minimization (ERM) – of finding the function in $\mathcal{H}$ which minimizes

$$\frac{1}{m}\sum_{i=1}^{m}(f(x_i)-y_i)^2$$

which is in general *ill-posed*, depending on the choice of the hypothesis space $\mathcal{H}$. Following Tikhonov (see for instance [6]) we minimize, instead, over the hypothesis space $\mathcal{H}_K$, for a fixed positive parameter γ, the regularized functional

$$\frac{1}{m}\sum_{i=1}^{m}(y_i - f(x_i))^2 + \gamma\|f\|_K^2, \tag{4}$$

where $\|f\|_K^2$ is the norm in $\mathcal{H}_K$ – the Reproducing Kernel Hilbert Space (RKHS), defined by the kernel K. The last term in Equation 4 – called *regularizer* – forces, as we will see, smoothness and uniqueness of the solution.

[1] The very closely related Support Vector Machine (SVM) classifier was used for the same family of applications, and in particular for bioinformatics and for face recognition and car and pedestrian detection [1].

Let us first define the norm $\|f\|_K^2$. Consider the space of the linear span of $K_{\overline{x}_j}$. We use $\overline{x}_j$ to emphasize that the elements of X used in this construction do not have anything to do *in general* with the training set $(x_i)_{i=1}^m$. Define an inner product in this space by setting $\langle K_x, K_{\overline{x}_j}\rangle = K(x, \overline{x}_j)$ and extend linearly to $\sum_{j=1}^r a_j K_{\overline{x}_j}$. The completion of the space in the associated norm is the RKHS, that is a Hilbert space $\mathcal{H}_K$ with the norm $\|f\|_K^2$ (see [3]). Note that $\langle f, K_x\rangle = f(x)$ for $f \in \mathcal{H}_K$ (just let $f = K_{\overline{x}_j}$ and extend linearly).

To minimize the functional in Equation 4 we take the functional derivative with respect to f, apply it to an element $\overline{f}$ of the RKHS and set it equal to 0. We obtain

$$\frac{1}{m}\sum_{i=1}^{m}(y_i - f(x_i))\overline{f}(x_i) - \gamma\langle f, \overline{f}\rangle = 0. \tag{5}$$

Equation 5 must be valid for any $\overline{f}$. In particular, setting $\overline{f} = K_x$ gives

$$f(x) = \sum_{i=1}^{m} c_i K_{x_i}(x) \tag{6}$$

where

$$c_i = \frac{y_i - f(x_i)}{m\gamma} \tag{7}$$

since $\langle f, K_x\rangle = f(x)$. Equation 3 then follows, by substituting Equation 6 into Equation 7.

Notice also that essentially the same derivation for a generic loss function $V(y, f(x))$, instead of $(f(x) - y)^2$, yields the same Equation 6, but Equation 3 is now different and, in general, nonlinear, depending on the form of V. In particular, the popular Support Vector Machine (SVM) regression and SVM classification algorithms correspond to special choices of non-quadratic V, one to provide a ′robust″ measure of error and the other to approximate the ideal loss function corresponding to binary (miss)classification. In both cases, the solution is still of the same form of Equation 6 for any choice of the kernel K. The coefficients c_i are not given anymore by Equations 7 but must be found solving a quadratic programming problem.

3 Theory

We give some further justification of the algorithm by sketching very briefly its foundations in some basic ideas of learning theory.

Here the data $(x_i, y_i)_{i=1}^m$ is supposed random, so that there is an unknown probability measure ρ on the product space $X \times Y$ from which the data is drawn.

This measure ρ defines a function

$$f_\rho : X \to Y \tag{8}$$

satisfying $f_\rho(x) = \int y d\rho_x$, where ρ_x is the conditional measure on $x \times Y$.

From this construction f_ρ can be said to be the true input-output function reflecting the environment which produces the data. Thus a measurement of *the error* of f is

$$\int_X (f - f_\rho)^2 d\rho_X \tag{9}$$

where ρ_X is the measure on X induced by ρ (sometimes called the *marginal measure*).

The goal of learning theory might be said to "find" f minimizing this error. Now to search for such an f, it is important to have a space $\mathcal{H}$ – the *hypothesis space* – in which to work ("learning does not take place in a vacuum"). Thus consider a convex space of continuous functions $f : X \to Y$, (remember $Y \subset \mathrm{I\!R}$) which as a subset of $C(X)$ is *compact*, and where $C(X)$ is the Banach space of continuous functions with $||f|| = \max_X |f(x)|$.

A basic example is

$$\mathcal{H} = \overline{I_K(B_R)} \tag{10}$$

where $I_K : \mathcal{H}_K \to C(X)$ is the inclusion and B_R is the ball of radius R in $\mathcal{H}_K$.

Starting from the data $(x_i, y_i)_{i=1}^m = z$ one may minimize $\frac{1}{m}\sum_{i=1}^m (f(x_i) - y_i)^2$ over $f \in \mathcal{H}$ to obtain a unique hypothesis $f_z : X \to Y$. This f_z is called the empirical optimum and we may focus on the problem of estimating

$$\int_X (f_z - f_\rho)^2 d\rho_X \tag{11}$$

It is useful towards this end to break the problem into steps by defining a "true optimum" $f_{\mathcal{H}}$ relative to $\mathcal{H}$, by taking the minimum over $\mathcal{H}$ of $\int_X (f - f_\rho)^2$. Thus we may exhibit

$$\int_X (f_z - f_\rho)^2 = S(z, \mathcal{H}) + \int_X (f_{\mathcal{H}} - f_\rho)^2 = S(z, \mathcal{H}) + A(\mathcal{H}) \tag{12}$$

where

$$S(z, \mathcal{H}) = \int_X (f_z - f_\rho)^2 - \int_X (f_\mathcal{H} - f_\rho)^2 \tag{13}$$

The first term, (S) on the right in Equation 12 must be estimated in probability over z and the estimate is called the *sample errror* (sometime also the *estimation error*). It is naturally studied in the theory of probability and of empirical processes. The second term (A) is dealt with via approximation theory (see for a review [5] and also [1]) and is called the *approximation error*. The decomposition of Equation 12 is related to the well known bias (A) and variance (S) decomposition in statistics.

3.1 Sample Error

First consider an estimate for the sample error, which will have the form:

$$S(z, \mathcal{H}) \leq \epsilon \tag{14}$$

with high confidence, this confidence depending on ϵ and on the sample size m.

Let us be more precise. Recall that the covering number or Cov#$(\mathcal{H}, \eta)$ is the number of balls in $\mathcal{H}$ of radius η needed to cover $\mathcal{H}$.

Theorem 3.1 *Suppose* $|f(x) - y| \leq M$ *for all* $f \in \mathcal{H}$ *for almost all* $(x, y) \in X \times Y$. *Then*

$$Prob_{z \in (X \times Y)^m} \{S(z, \mathcal{H}) \leq \epsilon\} \leq 1 - \delta$$

where $\delta = Cov\#(\mathcal{H}, \frac{\epsilon}{24M}) e^{-\frac{m\epsilon}{288M^2}}$.

The result is Theorem C^* of [3], but earlier versions (usually without a topology on $\mathcal{H}$) have been proved by others, especially Vapnik, who formulated the notion of VC dimension to measure the complexity of the hypothesis space for the case of $\{0, 1\}$ functions.

In a typical situation of Theorem 3.1 the hypothesis space $\mathcal{H}$ is taken to be as in Equation 10, where B_R is the ball of radius R in a Reproducing Kernel Hilbert Space (RKHS) with a smooth K (or in a Sobolev space). In this context, R plays an analogous role to VC dimension[12]. Estimates for the covering numbers in these cases were provided by Cucker, Smale and Zhou.

The proof of Theorem 3.1 starts from Hoeffding inequality (which can be regarded as an exponential version of Chebyshev's inequality of probability theory). One applies this estimate to the function $X \times Y \to \mathbb{R}$ which takes (x, y) to $(f(x) - y)^2$. Then extending the estimate to the set of $f \in \mathcal{H}$ introduces the covering number into the picture. With a little more work, theorem 3.1 is obtained.

3.2 Approximation Error

The approximation error $\int_X (f_{\mathcal{H}} - f_\rho)^2$ may be studied as follows.

Suppose $B : L^2 \to L^2$ is a compact, strictly positive (selfadjoint) operator. Then let $\mathbf{E}$ be the Hilbert space

$$\{g \in L^2, \, \|B^{-s}g\| < \infty\}$$

with inner product $\langle g, h\rangle_{\mathbf{E}} = \langle B^{-s}g, B^{-s}h\rangle_{L^2}$. Suppose moreover that $\mathbf{E} \to L^2$ factors as $\mathbf{E} \to C(X) \to L^2$ with the inclusion $J_{\mathbf{E}} : \mathbf{E} \hookrightarrow C(X)$ well defined and compact.

Let $\mathcal{H}$ be $\overline{J_{\mathbf{E}}(B_R)}$ when B_R is the ball of radius R in $\mathbf{E}$. A theorem on the approximation error is

Theorem 3.2 *Let $0 < r < s$ and $\mathcal{H}$ be as above. Then*

$$\|f_\rho - f_{\mathcal{H}}\|^2 \leq (\frac{1}{R})^{\frac{2r}{s-r}} \|B^{-r}f_\rho\|^{\frac{2s}{s-r}}$$

We now use $\|\cdot\|$ for the norm in the space of square integrable functions on X, with measure ρ_X. For our main example of RKHS, take $B = L_K^{1/2}$, where K is a Mercer kernel and

$$L_K f(x) = \int_X f(x')K(x, x') \qquad (15)$$

and we have taken the square root of the operator L_K. In this case $\mathbf{E}$ is $\mathcal{H}_K$ as above.

Details and proofs may be found in [3] and in [11].

3.3 Sample and approximation error for the regularization algorithm

The previous discussion depends upon a compact hypothesis space $\mathcal{H}$ from which the experimental optimum f_z and the true optimum $f_{\mathcal{H}}$ are taken. In the key algorithm of section 2 , the optimization is done over all $f \in \mathcal{H}_K$ with a regularized error function. The error analysis of sections 3.1 and 3.2 must therefore be extended.

Thus let $f_{\gamma,z}$ be the empirical optimum for the regularized problem as exhibited in Equation 4

$$\frac{1}{m}\sum_{i=1}^{m}(y_i - f(x_i))^2 + \gamma\|f\|_K^2. \tag{16}$$

Then

$$\int (f_{\gamma,z} - f_\rho)^2 \leq S(\gamma) + A(\gamma) \tag{17}$$

where $A(\gamma)$ (the approximation error in this context) is

$$A(\gamma) = \gamma^{1/2}\|L_K^{-\frac{1}{4}} f_\rho\|^2 \tag{18}$$

and the sample error is

$$S(\gamma) = \frac{32M^2(\gamma + C)^2}{\gamma^2} v^*(m, \delta) \tag{19}$$

where $v^*(m, \delta)$ is the unique solution of

$$\frac{m}{4}v^3 - ln(\frac{4m}{\delta})v - c_1 = 0. \tag{20}$$

Here $C, c_1 > 0$ depend only on X and K. For the proof one reduces to the case of compact $\mathcal{H}$ and applies theorems 3.1 and 3.2. Thus finding the optimal solution is equivalent to finding the best tradeoff between A and S for a given m. In our case, this bias-variance problem is to minimize $S(\gamma) + A(\gamma)$ over $\gamma > 0$. There is a unique solution – a best γ – for the choice in Equation 4. For this result and its consequences see [4].

4 Remarks

The tradeoff between sample complexity and hypothesis space complexity

For a given, fixed hypothesis space $\mathcal{H}$ only the sample error component of the *error* of f_z can be be controlled (in Equation 12 only $S(z, \mathcal{H})$ depends on the data). In this view, convergence of S to zero as the number of data increases (theorem 3.1) is then the central problem in learning. Vapnik called consistency of ERM (eg convergence of the empirical error to the true error) the key problem in learning theory and in fact much modern work has focused on refining the necessary and sufficient conditions for consistency of ERM (the uniform Glivenko-Cantelli property of $\mathcal{H}$, finite V_γ dimension for $\gamma > 0$ etc., see [6]). More generally, however, there is a tradeoff between minimizing the sample error and minimizing the approximation error – what we referred to as the bias-variance problem. Increasing the number of data points m decreases the sample error. The effect of increasing the complexity of the hypothesis space is trickier. Usually the approximation error decreases but the sample error increases. This means that there is an optimal complexity of the hypothesis space for a given number of training data. In the case of the regularization algorithm described in this paper this tradeoff corresponds to an optimum value for γ as studied by [4, 8]. In empirical work, the optimum value is often found through crossvalidation techniques [14].

This tradeoff between approximation error and sample error is probably the most critical issue in determining good performance on a given problem. The class of regularization algorithms, such as Equation 4, shows clearly that it is also a tradeoff – quoting Girosi – between the *curse of dimensionality* (not enough examples) and *the blessing of smoothness* (which decreases the effective "dimensionality" eg the complexity of the hypothesis space) through the parameter γ.

The regularization algorithm and Support Vector Machines

There is nothing to stop us from using the algorithm we described in this paper – that is square loss regularization – for *binary classification*. Whereas SVM classification arose from using – *with binary y* – the loss function

$$V(f(\mathbf{x}, y)) = (1 - yf(\mathbf{x}))_+,$$

we can perform least-squares regularized classification via the loss function

$$V(f(\mathbf{x}, y)) = (f(\mathbf{x}) - y)^2.$$

This classification scheme was used at least as early as 1989 (for reviews see [9] and then rediscovered again by many others (see [1]), including Mangasarian (who refers to square loss regularization as "proximal vector machines") and Suykens (who uses the name "least square SVMs"). Rifkin ([10]) has confirmed the interesting empirical results by Mangasarian and Suykens: "classical" square loss regularization works well also for binary classification (examples are in tables 1 and 2).

In references to supervised learning the Support Vector Machine method is often described (see for instance a recent issue of the Notices of the AMS) according to the "traditional" approach, introduced by Vapnik and followed by almost everybody else. In this approach, one starts with the concepts of *separating hyperplanes* and *margin*. Given the data, one searches for the linear hyperplane that separates the positive and the negative examples, assumed to be linearly separable, with the largest margin (the margin is defined as the distance from the hyperplane to the nearest example). Most articles and books follow this approach, go from the separable to the non-separable case and use a so-called "kernel trick" (!) to extend it to the nonlinear case. SVM for classification was introduced by Vapnik in the linear, separable case in terms of

	800		250		100		30	
	SVM	RLSC	SVM	RLSC	SVM	RLSC	SVM	RLSC
	0.131	0.129	0.167	0.165	0.214	0.211	0.311	0.309

Table 1: *A comparison of SVM and RLSC (Regularized Least Squares Classification) accuracy on a multiclass classification task (the* `20newsgroups` *dataset with* 20 *classes and high dimensionality, around* $50,000$*), performed using the standard "one vs. all" scheme based on the use of binary classifiers. The top row indicates the number of documents/class used for training. Entries in the table are the fraction of misclassified documents. From [10].*

	52		20		10		3	
	SVM	RLSC	SVM	RLSC	SVM	RLSC	SVM	RLSC
	0.072	0.066	0.176	0.169	0.341	0.335	0.650	0.648

Table 2: *A comparison of SVM and RLSC accuracy on another multiclass classification task (the* `sector105` *dataset, consisting of* 105 *classes with dimensionality about* $50,000$*). The top row indicates the number of documents/class used for training. Entries in the table are the fraction of misclassified documents. From [10].*

maximizing the margin. In the non-separable case, the margin motivation loses most of its meaning. A more general and simpler framework for deriving SVM algorithms for classification and regression is to regard them as special cases of regularization and follow the treatment of section 2. In the case of linear functions $f(x) = w \cdot x$ and separable data, maximizing the margin is exactly equivalent to maximizing $\frac{1}{||w||}$, which is in turn equivalent to minimizing $||w||^2$, which corresponds to minimizing the RKHS norm.

The regularization algorithm and learning theory

The Mercer theorem was introduced in learning theory by Vapnik; RKHS were explicitly introduced in learning theory by Girosi and later by Vapnik [1, 12]. Poggio and Girosi [9, 1] had introduced Tikhonov regularization in learning theory. Earlier, Gaussian Radial Basis Functions were proposed as an alternative to neural networks by Broomhead and Loewe. Of course, RKHS had been pioneered by Parzen and Wahba ([1]; for a review see [14]) for applications closely related to learning, including data smoothing (for image processing and computer vision, see [1]).

A Bayesian interpretation

The learning algorithm Equation 4 has an interesting Bayesian interpretation [14]: the data term – that is the first term with the quadratic loss function – is a model of (Gaussian, additive) noise and the RKHS norm (the stabilizer) corresponds to a prior probability on the hypothesis space $\mathcal{H}$.

Let us define $P[f|S_m]$ as the conditional probability of the function f given the training examples $S_m = (x_i, y_i)_{i=1}^m$, $P[S_m|f]$ as the conditional probability of

S_m given f, i.e. a model of the noise, and $P[f]$ as the *a priori* probability of the random field f. Then Bayes theorem provides the posterior distribution as

$$P[f|S_m] = \frac{P[S_m|f]\;P[f]}{P(S_m)}.$$

If the noise is normally distributed with variance σ, then the probability $P[S_m|f]$ is

$$P[S_m|f] = \frac{1}{Z_L} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{m}(y_i - f(x_i))^2}$$

where Z_L is a normalization constant.

If $P[f] = \frac{1}{Z_r} e^{-\|f\|_K^2}$ where Z_r is another normalization constant, then

$$P[f|S_m] = \frac{1}{Z_D Z_L Z_r} e^{-\left(\frac{1}{2\sigma^2}\sum_{i=1}^{m}(y_i - f(x_i))^2 + \|f\|_K^2\right)}.$$

One of the several possible estimates of f from $P[f|S_m]$ is the so called *Maximum A Posteriori* (MAP) estimate, that is

$$\max P[f|S_m] = \min \sum_{i=1}^{m} (y_i - f(\mathbf{x}_i))^2 + 2\sigma^2 \|f\|_K^2 \; .$$

which is the same as the regularization functional, if $\lambda = 2\sigma^2/m$ (for details and extensions to models of non Gaussian noise and different loss functions see [6]).

Necessary and sufficient conditions for learnability

Compactness of the hypothesis space $\mathcal{H}$ is *sufficient* for consistency of ERM, that is for bounds of the type of Theorem 3.1 on the sample error. The *necessary and sufficient* condition is that $\mathcal{H}$ is a uniform Glivenko-Cantelli class of functions, in which case no specific topology is assumed for $\mathcal{H}$[2]. There are several equivalent conditions on $\mathcal{H}$ such as finiteness of the V_γ dimension for all positive γ (which reduces to finiteness of the VC dimension for $\{0, 1\}$ functions) [3].

[2] *Definition: Let $\mathcal{F}$ be a class of functions f. $\mathcal{F}$ is a uniform Glivenko-Cantelli class if for every $\varepsilon > 0$*

$$\lim_{m\to\infty} \sup_{\rho} \mathbb{P}\{\sup_{f\in\mathcal{F}} |E_{\rho_m} f - E_\rho f| > \varepsilon\} = 0. \tag{21}$$

where ρ_n is the empirical measure supported on a set $x_1, ..., x_n$.

[3] In [2] – following [13] – a necessary and sufficient condition is proved for uniform convergence of $|I_{emp}[f] - I_{exp}[f]|$, in terms of the finiteness for all $\gamma > 0$ of a combinatorial quantity called V_γ dimension of $\mathcal{F}$ (which is the set $V(x), f(x), f \in \mathcal{H}$), under some assumptions on V. The result is based on a necessary and sufficient (distribution independent) condition proved by [13] which uses the metric entropy of $\mathcal{F}$ defined as $H_m(\epsilon, \mathcal{F}) = \sup_{x_m \in X^m} \log \mathcal{N}(\epsilon, \mathcal{F}, x_m)$, where $\mathcal{N}(\epsilon, \mathcal{F}, x_m)$ is the ϵ-covering of $\mathcal{F}$ wrt $l^\infty_{x_m}$ ($l^\infty_{x_m}$ is the l^∞ distance on the points x_m):

Theorem (Dudley, see [1]). $\mathcal{F}$ is a uniform Glivenko-Cantelli class iff $\lim_{m\to\infty} \frac{H_m(\epsilon,\mathcal{F})}{m} = 0$ *for all* $\epsilon > 0$.

We saw earlier that the regularization algorithm Equation 4 ensures (through the resulting compactness of the "effective" hypothesis space) well-posedness of the problem. It also yields convergence of the empirical error to the true error (eg bounds such as Theorem 3.1). An open question is whether there is a connection between well-posedness and consistency. For well-posedness the critical condition is usually stability of the solution. In the learning problem, this condition refers to stability of the solution of ERM with respect to small changes of the training set S_m. In a similar way, the condition number characterizes the stability of the solution of Equation 3. Is it possible that some specific form of stability may be necessary and sufficient for consistency of ERM? *Such a result would be surprising* because, a priori, there is no reason why there should be a connection between well-posedness and consistency: they are both important requirements for ERM but they seem quite different and independent of each other.

Learning theory, sample complexity and brains

The theory of supervised learning outlined in this paper and in the references has achieved a remarkable degree of completeness and of practical success in many applications. Within it, many interesting problems remain open and are a fertile ground for interesting and useful mathematics. One may also take a broader view and ask: what next?

One could argue that the most important aspect of intelligence and of the amazing performance of real brains is the ability to learn. How then do the learning machines we have described in the theory compare with brains? There are of course many aspects of biological learning that are not captured by the theory and several difficulties in making any comparison. One of the most obvious differences, however, is the ability of people and animals to learn from very few examples. The algorithms we have described can learn an object recognition task from a few thousand labeled images. This is a small number compared with the apparent dimensionality of the problem (thousands of pixels) but a child, or even a monkey, can learn the same task from just a few examples. Of course, evolution has probably done a part of the learning but so have we, when we choose for any given task an appropriate input representation for our learning machine. From this point of view, as Donald Geman has argued, the interesting limit is not "m goes to infinity," but rather "m goes to zero". Thus an important area for future theoretical and experimental work is learning from partially labeled examples (and the related area of active learning). In the first case there are only a small number ℓ of labeled pairs $(x_i, y_i)_{i=1}^{\ell}$ – for instance with y_i binary – and many unlabeled data $(x_i)_{\ell+1}^{m}$, $m >> \ell$. Though interesting work has begun in this direction, a satisfactory theory that provides conditions under which unlabeled data can be used is still lacking.

A comparison with real brains offers another, and probably related, challenge to learning theory. The "learning algorithms" we have described in this paper correspond to one-layer architectures. Are hierarchical architectures with more layers justifiable in terms of learning theory? It seems that the learning theory of the type we have outlined does not offer any general argument in favor of hierarchical learning machines for regression or classification. This is somewhat of a puzzle since the organization of cortex – for instance visual cortex – is strongly hierarchical. At the same time, hierarchical learning systems show superior performance in several engineering applications. For instance, a face categorization system in which a single SVM classifier combines the real-valued output of a few classifiers, each trained to a different component of faces – such as eye and nose –, outperforms a single classifier trained on full images of faces ([1]). The theoretical issues surrounding hierarchical systems of this type are wide open, and likely to be of paramount importance for the next major development of efficient classifiers in several application domains.

Why hierarchies? There may be reasons of *efficiency* – computational speed and use of computational resources. For instance, the lowest levels of the hierarchy may represent a dictionary of features that can be shared across multiple classification tasks (see [7]). Hierarchical system usually decompose a task in a series of simple computations at each level – often an advantage for fast implementations. There may also be the more fundamental issue of *sample complexity*. We mentioned that an obvious difference between our best classifiers and human learning is the number of examples required in tasks such as object detection. The theory described in this paper shows that the difficulty of a learning task depends on the size of the required hypothesis space. This complexity determines in turn how many training examples are needed to achieve a given level of generalization error. Thus the complexity of the hypothesis space sets the speed limit and the sample complexity for learning. If a task – like a visual recognition task – can be decomposed into low-complexity learning tasks, for each layer of a hierarchical learning machine, then each layer may require only a small number of training examples. Of course, not all classification tasks have a hierarchical representation. Roughly speaking, the issue is under which conditions a function of many variables can be approximated by a function of a small number of functions of subsets of the original variables. Neuroscience suggests that what humans can learn can be represented by hierarchies that are locally simple. Thus our ability of learning from just a few examples, and its limitations, may be related to the hierarchical architecture of cortex. This is just one of several possible connections, still to be characterized, between learning theory and the ultimate problem in natural science – the organization and the principles of higher brain functions.

Acknowledgments Thanks to Felipe Cucker, Federico Girosi, Don Glaser, Sayan Mukherjee, Martino Poggio and Ryan Rifkin.

References

[1] For a list of references, including specific applications mentioned in the article, see www.ai.mit.edu/projects/cbcl/projects/noticesams/poggiosmale.htm.

[2] N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *J. of the ACM*, 44(4):615–631, 1997.

[3] F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of AMS*, 39:1–49, 2001.

[4] F. Cucker and S. Smale. Best choices for regularization parameters in learning theory: on the bias-variance problem. *Foundations of Computational Mathematics*, 2(4):413–428, 2002.

[5] R.A. DeVore. Nonlinear approximation. *Acta Numerica*, 7:51–150, 1998.

[6] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer Verlag, Basel, 2001.

[8] 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.

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

[10] R. M. Rifkin. *Everything Old Is New Again: A Fresh Look at Historical Approaches to Machine Learning*. PhD thesis, Massachusetts Institute of Technology, 2002.

[11] S. Smale and D. Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1:1–25, 2003.

[12] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[13] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequences of events to their probabilities. *Th. Prob. and its Applications*, 17(2):264–280, 1971.

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