Information complexity of neural networks

Mark A. Kon[1], Boston University

and

Leszek Plaskota[2], Warsaw University

Addresses:

Department of Mathematics
Boston University
Boston, MA 02215
and
Department of Mathematics, Informatics, and Mechanics
Warsaw University
2 Banacha St.
02-097 Warsaw, Poland

Reprint requests to be sent to:

Prof. Mark A. Kon, Boston University, Dept. of Mathematics,
111 Cummington Street, Boston, MA, 02215,
tel: (617) 353 2560; fax: (617) 353 8100; email mkon@bu.edu

Running title: Information complexity

[1]Research partially supported by the National Science Foundation and the U.S. Fulbright Commission

[2]Research partially supported by the Polish-American Fulbright Foundation and the National Committee for Scientific Research of Poland

Abstract

This paper studies the question of lower bounds on numbers of neurons and examples necessary to program a given task into feedforward neural networks. We introduce the notion of information complexity of a network to complement that of neural complexity. Neural complexity deals with lower bounds for neural resources (numbers of neurons) needed by a network to perform a given task within a given tolerance. Information complexity measures lower bounds for information needed about (i.e., number of examples of) the desired input-output function. We study the interaction of the two complexities, and so lower bounds for the complexity of building and then programming feedforward nets for given tasks. We show something unexpected a priori - the interaction of the two can be simply bounded, so that they can be studied essentially independently. We construct order n^3 RBF algorithms, show that they are information-optimal, and give example applications.

Keywords: Feedforward neural network, complexity, information complexity, neural complexity, radial basis functions, RBF networks, learning

1. Introduction

Learning problems in feed-forward neural network theory are essentially partial information issues. That is, we wish to reconstruct a desired input-output (i-o) function from partial information consisting of examples (i.e., individual function evaluations). A complexity theory for neural networks from the standpoint of information complexity has had some beginnings in the work of Girosi & Poggio (1990), Poggio & Girosi (1989), and others.

The theory of neural complexity contrasts itself from that of information complexity in that the former deals with numbers of neurons in the hidden layer of a feed-forward network which are necessary for the computation of a given i-o function, generally assuming full information about that function within some tolerance ϵ. This theory has seen some extensive and successful development in recent years, particularly in the work, e.g., of Mhaskar (1996), Mhaskar &

Micchelli (1992, 1993, 1994, 1995), and Barron (1993). Additional work on function approximation issues has been done in Chui & Li (1992), and Chui, Li & Mhaskar (1993, 1996).

Information complexity is to an extent the "second half" of complexity theory for neural networks, that which deals with information issues, and numbers of examples needed to encode given tasks into neural networks. Neural complexity has been studied in the above references, while information complexity (the number of examples of an i-o function needed to approximate it within a given tolerance) is, we believe, still open to a good deal of development - in this paper we begin a study of the interaction of the two. We introduce a parameter k, the number of examples available in the construction of a network to complement the number n of hidden neurons available.

Just as neural complexity theory has its roots in classical approximation theory, the theory of information complexity is closely related to continuous complexity theory as currently studied (e.g., Traub, Wasilkowski & Woźniakowski, 1998). The proofs of many results in complexity of feed-forward neural nets, once the two sets of connections are established, reduce largely to developing results in or referencing work in these two areas, as occurs in several places here.

Initially we divide complexity of feed-forward nets into two exact scenarios. The first is when we know the i-o function f to be approximated exactly, i.e., we have unlimited information. The question here is, how complex a neural net (how many hidden units) do we need to express the function within a tolerance ϵ measured in some norm? The second arises when we assume unlimited neural resources (as large a hidden layer as we please), and ask how many examples of f are required for its approximation within ϵ. The question of what to do with limited information has been at the center of learning theory for a number of years, with algorithms for classical feedforward nets such as backpropagation and the Boltzmann machine having received a good deal of attention.

We will examine the second issue and then the combined question of what to do with limited numbers k of examples and n of neurons. The interaction of n and k is interesting, and we characterize this joint complexity's order. We show something unexpected a priori - that relationships between information and neural complexities can be simply bounded, and so the two issues can be studied independently before their interaction. The two complexity questions pose challenging problems in mathematics and neural phenomenology - it is in a sense fortunate they can be largely separated.

We wish to show these results have practical in addition to some theoretical significance. We believe they can be used directly in practical situations to obtain upper and lower bounds on numbers of examples and neurons needed to develop systems with desired i-o functions. We present some examples of how this might be accomplished. Beyond calculating complexities, we wish also to construct algorithms optimizing k and n, to show they are optimal or almost optimal, and to apply them to examples of interest. The prescriptions here, though they are mathematically optimal, are presented with useable algorithms which are of use in the construction of RBF networks.

We remark that the heart of the information complexity problem is the fact that the reconstruction of an i-o function f from incomplete (partial) and/or noisy information is an ill-posed problem. The best regularization techniques for this problem given natural a priori smoothness constraints on f can be shown equivalent to the use of optimal reconstruction algorithms in continuous complexity. It is remarkable that within the model of computation in which arbitrary hidden layer activation functions $G(\vec{x})$ are allowed, optimal algorithms for reconstructing f, i.e., algorithms which utilize information in examples with the greatest possible efficiency, are those using radial basis functions of the type studied by Mhaskar & Micchelli (1992), Micchelli & Buhmann (1992), Girosi & Poggio (1990), and Poggio & Girosi (1989, 1990), and in various works on continuous complexity, see e.g. Traub, Wasilkowski & Woźniakowski (1988), and Plaskota (1996).

More precisely, in the well-defined context of optimality given here, the algorithms are optimal in their use of information, assuming we have sufficiently many neurons in the hidden layer to work with (a number n at least equal to the number k of examples, see Theorem 2 in Section 6). Since the optimality results are not restricted to networks but are based on lower bounds of error for any system using the information in the examples, we show that no other network learning from examples can improve on this algorithm. We thus claim that with the computational provision of as many neurons as there are examples, along with the capability for basic linear algebra operations (essentially Gaussian elimination on a $k \times k$ system of equations) a particular RBF algorithm is a priori at least as accurate and efficient as backpropagation, the Boltzmann machine, or any other algorithm. When the neuron number n is limited and smaller than k, then our claim (Theorem 3) reduces to (almost) optimality only within the class of RBF neural networks with a given number of hidden units. Our error criteria are general to the extent they can involve essentially any norm.

We remark that the model of information complexity presented here is a continuous complexity model in which there is a separation between information and algorithmic complexity. It fits a widely used template for so-called standard information $Nf = (f(\vec{x}_1), f(\vec{x}_2), \ldots, f(\vec{x}_k))$, i.e., the desired output values $f(\vec{x}_i)$ at a series of examples (input vectors) $\vec{x}_i$. The algorithmic portion ϕ (which computes the coefficients of the approximation network from information Nf) fits a model of computation in which each additional neuron has a unit cost, and remaining computations are deemed negligible in their contributions to complexity. Thus the number k of examples represents information complexity, and the number n of neurons (i.e., RBF's) used represents algorithmic complexity in this model. Since the latter corresponds to the number of neurons, we call it neural complexity. We show algorithms using radial basis functions are optimal from the standpoint of information complexity in that for a given number k of examples, these algorithms provide the smallest possible error.

Using the above notation, the full neural system computes a function $\phi(Nf)$, with the algorithmic portion ϕ using information Nf from examples to compute the network approximation of f. Thus letting F_1 be the class of functions from which f is drawn, $N(F_1) \subset \mathbb{R}^k$ be the space of possible information $(f(\vec{x}_1), \ldots, f(\vec{x}_k))$, and I be the identity operator $If = f$, we wish to approximate the identity operator with our neural network, which is a composition $\phi \circ N$ of the information operation N with the algorithmic operation ϕ.

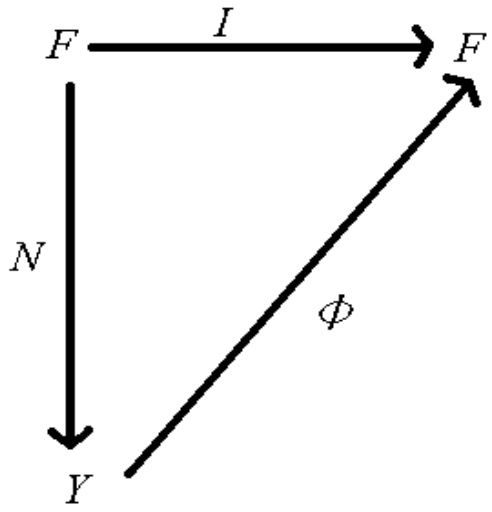


Fig. 1: The schematic relation of the spaces in continuous complexity theory

We note in passing there are other useful ways of measuring complexity suited to other requirements. For example, the complexity of *computing* the weights of the neural network which best approximate the desired i-o function (a linear algebra operation here), is not part of our computational model, though this can be changed if needed. Though the algorithms presented here do not necessarily perform the task of finding the optimal weights in the fastest way, they are quite tractable from a practical standpoint for problems whose essential input dimension is less than 100, and optimal in the number of examples required for a given error tolerance.

A main theme of this paper is the remarkable set of parallels between the fundamentals of feed-forward neural network theory and two very established areas of mathematics and theoretical computer science, namely approximation theory and continuous complexity theory. The parallel with the former has been seen in a number of works on neural complexity (see above references), while a parallel with the latter is studied here, in the connection between neural information complexity and continuous complexity. Some mathematical statements on approximation optimality translate directly into statements on optimality of neural networks, and one purpose of this paper is to show how this theory can be applied to answer our questions. The brevity of our proofs indicates how well-developed continuous complexity is, and the extent to which the present information complexity theory depends on it.

In this paper, we consider only the deterministic, worst case setting. There may be a need for more theoretical study of other settings such as average case setting and ones in which information is taken at random. In these cases there is also a complexity theory (Traub, Wasilkowski & Woźniakowski, 1988, Plaskota, 1996), and we believe that similar results can be obtained.

2. Definitions and main result.

We will generally assume that our input-output (i-o) function f is multivariate. That is,

$$f : D \to \mathbb{R},$$

where $D = \mathbb{R}^d$ or D is a proper subset of $\mathbb{R}^d$, e.g., $D = [0,1]^d$ is a d-dimensional unit cube. We have some a priori knowledge of f, e.g., that f is in a sense smooth. This is expressed by assuming

$$f \in F_1,$$

where F_1 is a ball of a normed space. This assumption is general in that any convex, balanced, and absorbing set is a unit ball of such a space.

We wish to construct a neural network approximation of f. An approximation is constructed based only on this a priori information and on a posteriori information about f given in examples

$$y_i = f(\vec{x}_i)$$

for $1 \leq i \leq k$. We consider three layer neural networks with node activations x_i in the first (input) layer, r_i in the second (hidden) layer, and q for the single output neuron in the third layer.

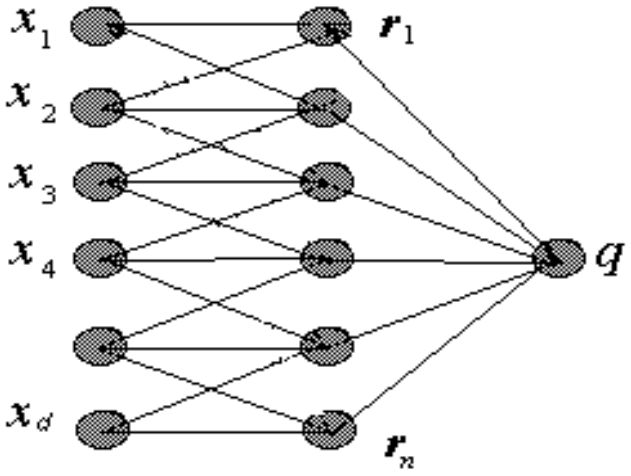


Fig. 2: Model three layer network

(We remark that the general case of more than one neuron in the output layer can be handled as a combination of the single output neuron models we discuss here.)

We assume an RBF model, by which we mean that activations of neurons in the hidden layer are given by $r_i = G_i(\vec{x})$ for given functions G_i, where $\vec{x} = (x_1, \ldots, x_d)^T$ (T is transpose), while the output q is linear in the activations of the second layer, i.e.,

$$q(\vec{x}) = \sum_{i=1}^{n} w_i r_i = \sum_{i=1}^{n} w_i G_i(\vec{x}). \tag{1}$$

Here n is the number of neurons in the hidden layer.

We now state the main results of the paper. More detailed definitions and statements are left to the body of the paper. We assume for the results that our function space is a *reproducing kernel Hilbert space* (see below). Essentially this is a Hilbert space in which function evaluation is a well-defined, i.e., continuous, operation.

Let H be the reproducing kernel Hilbert space with reproducing kernel $G : D \times D \to \mathbb{R}$. Suppose our a priori information places f in the ball

$$F_1 = \{\ f \in H : \|f\|_H \leq a\ \}$$

of H. We approximate f by an RBF neural network q of the form (1) with activation functions $G_i(\,\cdot\,) = G(\vec{x}_i,\,\cdot\,)$, for some $\vec{x}_i \in D$ (see Section 6).

By an *optimal* algorithm for constructing network approximations we mean the one which minimizes the worst case error of approximation over all possible f from F_1. We also mention that an *almost optimal* algorithm ϕ is one whose worst case error is within a (usually small) fixed factor C of the minimal worst case error.

Theorem I. *Suppose the number n of available neurons is not smaller than the number k of examples. An optimal algorithm (network) for approximating f from information $\vec{y} = Nf = (f(\vec{t}_1), \ldots, f(\vec{t}_k))^T$ is given by the linear combination*

$$q_{\vec{y}} = \sum_{j=1}^{k} c_j G(\vec{t}_j,\,\cdot\,),$$

where the coefficients $\vec{c} = (c_1, \ldots, c_k)^T$ are solutions of the linear system $M_N \vec{c} = \vec{y}$ with the matrix $M_N = (G(\vec{t}_i, \vec{t}_j))_{i,j=1}^{k}$.

Theorem II. *Let Al be an almost optimal algorithm that gives neural network approximations with a limited number n of neurons and assuming full information about a function. Then the composite algorithm $\vec{y} \mapsto q_{\vec{y}} \mapsto Al(q_{\vec{y}})$ is almost optimal for approximating f by an n neuron network from information $\vec{y} = N(f)$ about f.*

The *neural complexity* $n(\epsilon)$, is the minimal number of neurons in the hidden layer sufficient to approximate any f from the ball F_1 with accuracy ϵ assuming full information about f. The *information complexity* $k(\epsilon)$, is the minimal number of examples $(\vec{x}_i, f(\vec{x}_i))$ from which it is possible to approximate f within ϵ, assuming an unlimited number of neurons available.

Theorem III. *Information complexity dominates neural complexity, i.e.,*

$$n(\epsilon) \le k(\epsilon), \qquad \forall \epsilon > 0.$$

Theorem IV. *In order to approximate any function f from the ball $\|f\|_H \leq a$ with error $\epsilon > 0$, it is necessary to use at least $n(\epsilon)$ neurons and $k(\epsilon)$ examples, and sufficient to use at most $n(\epsilon/2)$ neurons and $k(\epsilon/2)$ examples.*

3. Examples

As a concrete example to motivate this work, suppose we are building a control system in which homeostatic parameters such as temperature, humidity, and specific chemical contents of an industrial mixture can be controlled (i.e., are input variables), and the output of these inputs, the ratio of elasticity and strength of a plastic which is produced from the mixture, is also recorded. Theoretically, combinations of values of the input variables may have unpredictable effects on the output variables, including binary and tertiary correlations of input variables, as well as even more complicated dependences. We wish to build a neural network whose input is the vector $\vec{x} = (x_1, \ldots, x_d)$ of input parameters, and whose output $q = f(\vec{x})$ is the ratio of elasticity and strength. The function f is unknown, and we have experimental data of the form $\{(\vec{x}_i, f(\vec{x}_i))\}_{i=1}^k$ from many previous runs of our equipment. We wish to build a network which will run on a moderate size computer and which will correctly predict the elasticity/strength ratio from our homeostatic parameters. Because of limits on our computational equipment, we must limit the size n of our network, and because of limited data, k is also limited. If we specify an error tolerance ϵ for this ratio, we will wish to optimize our procedure for constructing our ϵ-network by optimizing some function $h(n, k)$, which may depend on just n, just k, a linear combination of the two (with weights determined by the relative difficulty of increasing computational scale versus obtaining information), or a more complicated function. This problem can be somewhat more tamely stated if we invert the dependence of k, n on ϵ. This leads to a question which combines complexity Questions 1 and 2 below, and is stated in Question 4.

A second example (modified from one which was worked on by a consultant known to the authors) consists of a neural network which studies purchasing patterns of people using various mail order corporations. Corporations share data on consumers, and correct "mining" of such data can produce large numbers of sales to clients very likely to purchase given classes of products. In this case, a reasonable approach is to create a tree structure on the family of products under consideration (e.g., a node would be appliances, while a subnode would be

kitchen appliances, under which would be ovens, etc.), and as input variables for a given individual to include the dollar quantities x_i of purchases of the given consumer. The desired output in this case would be $f(\vec{x})$, the probability that the consumer would purchase the present target product, say a blender, toaster, or oven. In this situation we have a large dimension d of the input data (i.e., there are many products which can be purchased), as well as unchangeable size k of the learning set $\{(\vec{x}, f(\vec{x}))\}$, and we are interested here in finding the algorithm yielding the network with smallest practical ϵ for our given k and computationally limiting n. Question 4 is at the heart of this problem as well.

We remark that the assumptions we have made here regarding assumed membership of the unknown i-o function in a ball of a function space, e.g., L_s^∞ or another Sobolev space, are reasonable if a function with small norm in the said space can be assumed to approximate pointwise the i-o function within a "good" tolerance. It is unnecessary that the unknown function be exactly smooth, or exactly belong to the indicated class. There must be a global fit which is acceptable, and under such assumptions these theorems can be applied.

Suppose we again have the previously mentioned homeostatic system, and know that the variation of the output quality is such that the quality can be approximated by a 10 times differentiable function (e.g., a polynomial) whose first 10 derivatives are smaller than 15 (in some appropriate scale). In the case of polynomial approximation, this would place bounds on the coefficients of the polynomials, a reasonable way to "guess" the nature of a function approximating an unknown one. (Though of course such bounds would have to be considered a heuristic process, techniques and guidelines for such approximate bounding can be established; see below). The above assumption would imply that the i-o function can be well-approximated by a function in the Sobolev space L_{10}^∞, in the ball of radius 15. This would provide our candidate function set F_{15} (the ball of radius 15 in $F = L_{10}^\infty$).

Next, with this information plus a given error tolerance, say $\epsilon = .1$ (in possibly another error metric, say the L^2 metric), we could compute theoretical upper and lower bounds for the neural and information complexities of our problem using theorems such as Theorem 1 and standard techniques in continuous complexity theory (Traub, Wasilkowski & Woźniakowski, 1988). This does not in itself say what examples to use and how to program the weights for an optimal such network, but it has been shown that using sparse grid points is very effective in such a regard, and that the algorithm of Theorem I is optimal in finding appropriate weights. Thus it would be practically feasible not only to

use the minimum numbers of neurons and examples to solve this practical problem, but also to program the weights of the network which results.

More specifically, such a computation might show that with full information we would need 1,000 neurons for error $\epsilon/2 = .05$, while with unlimited numbers of neurons we would need at least 2,000 data points for error $\epsilon/2 = .05$ (via informational techniques explicitly given in Traub, Wasilkowski & Woźniakowski, 1988). Note that by Theorem III the complexity functions $n(\epsilon)$ and $k(\epsilon)$ satisfy $n(\epsilon) \leq k(\epsilon)$. Then we would use Theorem IV to estimate that we are guaranteed error $\epsilon = .1$ or less with 1,000 neurons and 2,000 data points.

Experimental verification of the above bounds on polynomial approximations could be made after data are taken. Namely, derivatives of such approximating smooth functions can be calculated, and in a bootstrapping way used to verify (or negate) the underlying assumptions about membership by the function f in the set F_{15}. For example, that the gradient of the data is bounded by, say, 5 units can easily be verified from our algorithmic approximation or experimentally from the data, and higher derivatives can be bounded similarly.

4. Neural complexity ($k = \infty$)

We now formally consider the above questions, and their answers in our mathematical context.

Question 1. *Given a fully known i-o function f and an error tolerance ϵ, what is the smallest size n of the hidden layer in a network that can approximate f within ϵ?*

This involves a complete information setting, and is partially answered in, e.g., the work of Barron (1993), Chui & Li (1992), Micchelli & Mhaskar (1992, 1993, 1994, 1995), as well as others. Approximation results depend on the space of functions being considered, as well as the family of approximating functions allowed.

We present a result of Micchelli & Mhaskar (1993) for a particular space of functions, under the assumption that arbitrary dilations and translations of a given periodic function p are allowed. This theorem is a typical consequence of the work done in neural complexity.

Let C^{d*} denote the space of 2π-periodic continuous functions on $\mathbb{R}^d$. For $f \in C^{d*}$, define $E_n^d(f) \equiv \min_{P \in T_n} \|f - P\|$, where T_n is the set of trigonometric polynomials of order n or less in C^{d*} and $\|\cdot\|$ is the supremum norm.

Thus $E_n^d(f)$ represents the best possible error with n terms of a Fourier series. Let $S_p \subseteq \{\boldsymbol{m} \in \mathbb{Z}^d : \widehat{p}(\boldsymbol{m}) \neq 0\}$ where $\widehat{p}(\boldsymbol{m})$ denotes Fourier series coefficients, and *assume* there is a set J containing $d \times s$ matrices with integer entries such that $\{A^T \boldsymbol{m} : \boldsymbol{m} \in S_p,\, A \in J\} = \mathbb{Z}^s$, where A^T denotes the transpose. If $d = s$ and p is a function with none of its Fourier coefficients equal to zero (the radial basis case), then we may choose $S_p = \mathbb{Z}^s$ and $J = \{I_{s\times s}\}$ the identity matrix. Let $\boldsymbol{k_m}$ be the multi-integer with minimum magnitude such that $\boldsymbol{m} = A^T \boldsymbol{k_m}$ for some $A = A_{\boldsymbol{m}} \in J$, and $N_n = \max\{|\boldsymbol{k_m}| : -2n \leq \boldsymbol{m} \leq 2n\}$. Let

$$m_n \equiv \min\{|\widehat{p}(k_m)| : \ -2n \leq m \leq 2n\},$$

where the last inequality is taken componentwise in m. Then we have:

Theorem 1. *Let $s \geq d \geq 1$, $n \geq 1$, and $N \geq N_n$ be integers, $f \in C^{s*}$, $p \in C^{d*}$. There exists a network*

$$q(\vec{x}) = q_{n,N,p}(f;\vec{x}) \equiv \sum_j d_j p(A_j \vec{x} + \vec{t}_j) \tag{2}$$

such that

$$\|f - q_{n,N,p}(f)\| \leq c\left(E_n^s(f) + \frac{E_N^d(p) n^{s/2}}{m_n}\|f\|\right).$$

In (2) the sum contains at most $O(n^s N^d)$ terms, $A_j \in J$, $t_j \in \mathbb{R}^d$, and d_j are linear functionals of f, depending on n, N, p.

These types of full-information results are benchmarks (and lower bounds) for comparison with complexities in partial information settings. In our setting we can use such results directly to compute (or estimate) the neural complexity.

Definition 1. Given a family $\Gamma = \{G_\alpha\}_\alpha$ of activation functions, let $\mathcal{N}_n$ be the set of neural networks with n hidden neurons,

$$\mathcal{N}_n = \left\{ q = \sum_{i=1}^{n} w_i G_i : \quad G_i \in \Gamma, w_i \in \mathbb{R} \right\}.$$

For a function f and error norm $\|\cdot\|_e$, we define the nth error of approximation as $e(f,n) = \inf_{q\in\mathcal{N}_n} \|f-q\|_e$. The nth error over a class F_1 of functions is

$$e_1(n) = \sup_{f\in F_1} e(f,n) = \sup_{f\in F_1} \inf_{q\in\mathcal{N}_n} \|f-q\|_e.$$

Definition 2. The *local neural complexity* $n(f,\epsilon)$ of evaluating a function f is the smallest number n of neurons in the hidden layer required for the estimation of f within error ϵ. The *global neural complexity* (or neural complexity) on a class F_1 of functions is

$$n(\epsilon) \equiv \sup_{f\in F_1} n(f,\epsilon).$$

5. Information complexity ($n = \infty$)

Question 2. *Given an unknown function f in some class, what is the smallest number k of examples $\{(\vec{x}_i, f(\vec{x}_i))\}_{i=1}^k$ for which it is (theoretically) possible to estimate f within error ϵ (assuming unlimited access to hidden units)?*

In order to answer this we assume our prior knowledge of f places it into a set F_1. For instance, F_1 may be a convex, balanced, and absorbing set in a linear space F. This is known to determine uniquely a norm on this space, with respect to which the functions $f \in F_1$ form the unit ball (or more generally a ball of radius a). The restriction to a ball of F_1 is a natural consequence of the above basic assumption on F_1.

Definition 3. Information of the form

$$\vec{y} = Nf = (f(\vec{x}_1), f(\vec{x}_2), \ldots, f(\vec{x}_k))^T \in \mathbb{R}^k$$

is *standard information* about f. Information is *adaptive* if the choice of $\vec{x}_i$ depends on previously obtained values, i.e., $\vec{x}_i = \vec{x}_i(f(\vec{x}_1), \ldots, f(\vec{x}_{i-1}))$.

Otherwise information is *nonadaptive*. The *cardinality of information* is the number of examples in the information, i.e., $\text{card}(N) = k$.

We henceforth assume information Nf is standard unless otherwise specified. Then Question 2 can be formulated in the language of continuous complexity as follows. Given an unknown $f \in F_1$, what is the minimum number of examples (i.e., cardinality of information N) for which the error ϵ can be achieved, using the best possible algorithm for reconstructing f?

We denote the reconstruction algorithm by ϕ: $N(F_1) \to F$. Thus ϕ takes information $\vec{y}$ about f and produces an approximation $\phi(\vec{y}) \in F$. The error of approximation for f is defined as

$$e(\phi, N, f) = \|f - \phi(Nf)\|_e,$$

where $\|\cdot\|_e$ is a given error norm.

Definition 4. The (worst case) error of the algorithm ϕ using information N on the class F_1 is defined as

$$e(\phi, N) = \sup_{f \in F_1} e(\phi, N, f) = \sup_{f \in F_1} \|f - \phi(Nf)\|_e.$$

An algorithm ϕ^* is *optimal* in a class of algorithms iff it minimizes the worst case error.

Note in general a nonzero error is present due to the fact that our function f cannot be uniquely identified by information $\vec{y}$. Hence the same approximation $\phi(\vec{y})$ will correspond to all functions from the set

$$F(\vec{y}) \equiv N^{-1}(\vec{y}) \cap F_1$$

of functions consistent with information $\vec{y}$.

It is desirable that the algorithm not only minimize the worst case error over F_1, but also the worst case error over $F(\vec{y})$, for each $\vec{y} \in N(F_1)$. We remark (see e.g. Traub, Wasilkowski & Woźniakowski, 1988, for details) that this is satisfied iff $\phi(\vec{y})$ is just the (Chebyshev) center of $F(\vec{y})$. Such an algorithm, if it exists, is known as *central* and denoted by ϕ^c. Then the (Chebyshev) radius $r(F(\vec{y}))$ in the $\|\cdot\|_e$ norm (i.e., the radius of the smallest ball containing it) becomes the smallest possible error for information y,

$$e(N,\vec{y}) \equiv \inf_{\phi(\vec{y})} \sup_{f\in F(\vec{y})} \|f-\phi(\vec{y})\|_e = \|f-\phi^c(\vec{y})\|_e \qquad (3)$$
$$= r(F(\vec{y}))$$

The supremum

$$e(N) = \sup_{\vec{y}\in N(F_1)} e(N,\vec{y}) = \inf_{\phi:N(F_1)\to F} \sup_{f\in F_1} \|f-\phi(N(f))\|_e \qquad (4)$$
$$= \sup_{f\in F_1} \|f-\phi^c(N(f))\|_e$$
$$= \inf_{\phi:\mathbb{R}^k\to F} \sup_{f\in F_1} \|f-\phi(N(f))\|_e$$

is denoted as minimum worst case error for information N, also called the *radius of information,*

$$r(N) \equiv \sup_{f\in F_1} r(F(Nf)) = \sup_{f\in F_1} e(N,N(f)) = e(N), \qquad (5)$$

by (4). (Which of the several definitions of $e(\,\cdot\,)$ is used will be clear from the arguments.)

Definition 5. A family $\Gamma = \{G_\alpha\}_\alpha$ of activation functions for which the set $\widetilde{\mathcal{N}} = \bigcup_n \mathcal{N}_n$ of neural networks $q(\vec{x}) = \sum_{i=1}^{n} w_i G_i(\vec{x})$ (with arbitrary number n of neurons) is dense in F with respect to error norm $\|\cdot\|_e$, is called *complete.*

Thus if information is the only limitation and the family of Γ is complete (i.e., functions in F_1 can be reconstructed by networks with arbitrarily small error), then issues of function reconstruction reduce to geometric ones, involving radii of sets. In particular under such circumstances, the error of the best possible algorithm using information N for reconstructing f is the supremum of radii of slices of a set in a normed linear space (equation (5)).

The networks are complete, for example, when $G_\alpha(\vec{x}) = G(\vec{x}-\vec{\alpha})$ is the family induced by $G(\vec{x})$ such that its Fourier transform vanishes on a set of measure 0, guaranteeing by a theorem of Wiener that the translates of G are dense in L^2. In a more general case we may have $G_\alpha(\vec{x}) = G(\vec{x},\vec{\alpha})$, where $G(\,\cdot\,,\,\cdot\,)$ is a symmetric and positive definite function defined on a set $D\times D \subset \mathbb{R}^d$ (see Section 6). This family is complete in the reproducing kernel Hilbert space H induced by G, with respect to an error norm which is weaker than the norm of H.

Definition 6. We define the minimal error with information of cardinality k to be

$$e_2(k) \equiv \inf_{\text{card}(N)=k} e(N) = \inf_{\text{card}(N)=k} r(N) \equiv r(k).$$

Then the ϵ-*information complexity* of function approximation in the set F_1 is the smallest cardinality of information sufficient to obtain error ϵ or less,

$$k(\epsilon) = \inf \{j : \, e_2(j) \leq \epsilon\}.$$

Remark 1. The issues related to this reconstruction problem are discussed in depth in Traub, Wasilkowski & Woźniakowski (1988), and so details are omitted here. An interesting theorem in this regard relates explicit geometric quantities in the space F with error norm $\|\cdot\|_e$ we are considering, to information error of approximation. We define, for $A \subseteq F$ a balanced subset of F (i.e. one for which $g \in A \Rightarrow -g \in A$), the Gelfand k-width of A by

$$d^k = d^k(A, F) \equiv \inf_{A^k} \sup_{g \in A \cap A^k} \|g\|_e,$$

where A^k is a subspace of F of codimension at most k. Thus d^k is the diameter of the intersection of A with a hyperplane of codimension k, minimized over the choice of hyperplane. Furthermore let $e_2^*(k)$ be the corresponding minimal error of reconstruction obtained from information consisting of k arbitrary linear functionals L_i, $N(f) = (L_1(f), \ldots, L_k(f))$ (instead of standard information). Then

$$\frac{1}{2} d^k \leq e_2^*(k) \leq d^k$$

with $d^k = d^k(F_1, F)$. We obviously have $e_2^*(k) \leq e_2(k)$. We also often have the reverse inequality up to a constant independent of k. In this case, the minimal error of standard information is essentially the Gelfand k-width of the a priori set F_1 in the norm $\|\cdot\|_e$. Such k-widths are calculated for various spaces (see Pinkus, 1985, Traub, Wasilkowski & Woźniakowski, 1988), and so our main point of interest is the fact that Question 2 reduces to this well-studied and geometric theory.

A natural choice of F is a space of smoothed square integrable functions, say $H^s(a)$, the ball of radius a in the space

$$H^s = \{f \in L^2(I^d) : \|f\|_s < \infty\}$$

of functions defined, say, on the unit d dimensional cube $I^d = [0,1]^d$, with the norm $\|f\|_s^2 = \inf_{f^*} \int_{\mathbb{R}^d} dx(|f^*|^2 + |(-\Delta)^{s/2} f^*|^2)$, the infimum taken over all functions f^* on $\mathbb{R}^d$ with $f^*\Big|_{I^d} = f$.

Here $\Delta = \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}$. For s an even integer this class is contained in the class

$$W_s^2(a') = \left\{ f : I^d \to \mathbb{R} \Big| \sum_{|\alpha|=s} \|f^{(\alpha)}\|_2 \leq a' \right\},$$

with $\|g\|_2 = \left(\int_{I^d} g^2 dx \right)^{1/2}$, for some choice of a'. It is known that when the Hilbert space H in the above discussion is replaced by W_s^2, then the error $e_2(k)$ for functions of norm bounded by $\|f\| \leq a$ is bounded above by $ck^{-s/d+1/2}$ for some $c > 0$, as long as $2s > d$ (Traub, Wasilkowski & Woźniakowski, 1988, p. 138). Thus under this assumption, this is also an upper bound for functions in $H^s(a) \subset W_s^2(a')$. This illustrates the type of existing results in continuous complexity theory which apply to the $n = \infty$ study of information complexity.

6. Interaction of information and neural complexities.

Question 3. *Given information about f with k examples, what is the best network approximating f which uses at most n neurons in the hidden layer?*

This question is crucial for a more general one:

Question 4. *In Question 1 it is assumed that network size n is the only limitation (i.e. we have full information about i-o function f). In Question 2 it is assumed the number k of examples (i.e., the cardinality of information) is the only limitation. In practice both parameters are limited. How do the values of n and k interact in determining network error ϵ?*

Here we will be more specific about our setting than earlier. We assume the set F_1 to which our i-o function f belongs is a ball in a *reproducing kernel Hilbert space* H, i.e., $\|f\|_H \leq a$, for some fixed $a > 0$. Such an assumption is in fact natural from our standpoint, since such spaces are essentially defined by the condition that pointwise evaluations of functions in these spaces are continuous (and so well-defined) operations.

Definition 7. A Hilbert space H with the inner product $\langle \cdot\,, \cdot \rangle_H$ consisting of multivariate functions defined on $D \subset \mathbb{R}^d$ is a *reproducing kernel Hilbert space* if function evaluation $f \mapsto f(\vec{x})$ is a bounded linear functional for any $\vec{x} \in D$. A reproducing kernel of H is a symmetric and positive definite function $K(\vec{x}, \vec{y})$ such that for any $f \in H$ and $\vec{x} \in D$,

$$f(\vec{x}) = \langle f, K(\vec{x}, \cdot\,) \rangle_H. \tag{6}$$

Note that any r.k.h.s. H possesses its reproducing kernel and, moreover, H is uniquely determined by its reproducing kernel K, see e.g. Aronszain (1950).

We also assume that the family Γ of possible activation functions is determined by the reproducing kernel K of the space H. That is, $\Gamma = \{G_{\vec{\alpha}}\}_{\vec{\alpha} \in D}$ with

$$G_{\vec{\alpha}}(\vec{x}) = K(\vec{x}, \vec{\alpha}), \qquad \vec{x}, \vec{\alpha} \in D.$$

Example 1. Let H be the space of univariate functions $f : [0, 1] \to \mathbb{R}$ that are absolutely continuous and vanish at zero with norm

$$\|f\|_H = \sqrt{\int_0^1 (f'(x))^2 dx}.$$

Then the corresponding reproducing kernel is $G(x, y) = \min(x, y)$. In this case, the possible activation functions are given as $x \mapsto \min(x, u_j)$ with some $u_j \in [0, 1]$. The networks are of the form

$$q(x) = \sum_{j=1}^{n} c_j \min(x, u_j),$$

and it can be easily seen that $q(x)$ is continuous piecewise linear function (with knots u_j) vanishing at zero. The error here can be measured, e.g., in L^2 norm.

Note that L^2 is weaker than $\|\cdot\|_H$ and hence the set of activation functions is in this case complete.

Important examples of reproducing kernels are shift-invariant kernels, i.e., ones for which $G(\vec{x},\vec{y}) = G_1(\vec{y}-\vec{x})$ with some G_1. For instance, the Ornstein-Uhlenbeck kernel in $\mathbb{R}^d$ is determined by $G_1(\vec{x}) = \frac{1}{2}\exp(-\|\vec{x}\|_1)$, $\vec{x}\in\mathbb{R}^d$, with $\|\vec{x}\|_1 = \sum_{j=1}^{d}|x_j|$, $\vec{x} = (x_1,\ldots,x_d)^T$.

Definition 8. We denote by $e(n,N)$ the minimal error of approximation in F_1 using information N and a limited number n of neurons,

$$e(n,N) = \inf_{\phi:N(F_1)\to\mathcal{N}_n}\ \sup_{f\in F_1}\ \|f-\phi(Nf)\|_e.$$

Furthermore, we denote by $e(n,k)$ the minimal error of approximation using k examples and networks with n hidden neurons, i.e.,

$$e(n,k) = \inf\ e(n,N),$$

the infimum taken over all information N of cardinality k.

Observe first that any approximation with n neurons based on partial information cannot be better than that based on full information about f, and it cannot be better than approximation with unlimited number of neurons. Hence

$$e(n,N) \geq \max(\, e_1(n), r(N)\,),$$

$$e(n,k) \geq \max(\, e_1(n), e_2(k)\,).$$

We are now ready to state theorems about how to construct optimal networks. We will assume that information is nonadaptive, i.e., the same sample points t_i are used for all f's. This is justified by the fact that, for approximating functions in a ball of a Hilbert space, adaptive information is no better than nonadaptive information. In other words, information complexity can be realized by nonadaptive information, see e.g. Traub, Wasilkowski & Woźniakowski (1988, Chap. 4).

Theorem 2. *Suppose the number n of available neurons is not smaller than the number k of examples. For standard information*

$$\vec{y} = Nf = (f(\vec{t}_1), \ldots, f(\vec{t}_k))^T,$$

let the network $q_{\vec{y}}$ be given as

$$q_{\vec{y}} = \sum_{j=1}^{k} c_j G(\vec{t}_j, \cdot),$$

where the coefficients $\vec{c} = (c_1, \ldots, c_k)^T$ are solutions of the linear system $M_N \vec{c} = \vec{y}$ with the matrix $M_N = (G(\vec{t}_i, \vec{t}_j))_{i,j=1}^k$. Then the algorithm $\phi^*(\vec{y}) = q_{\vec{y}}$ is optimal and central, i.e., $e(\phi^*, N) = r(N) = e(n, N)$.

Proof. The network $q_{\vec{y}}$ is the H-orthogonal projection of f onto the subspace V spanned by the activation functions $G(\vec{t}_j, \cdot)$, $1 \le j \le k$. Indeed, since $G(\vec{t}_j, \cdot)$ is the representor of function evaluation at $\vec{t}_j$ in the space H (i.e., plays the role of K in equation (6)), we have

$$\langle\, (f - q_{\vec{y}}), G(\vec{t}_j, \cdot)\,\rangle_H = f(\vec{t}_j) - q_{\vec{y}}(\vec{t}_j) = f(\vec{t}_j) - y_j = 0,$$

and $f - q_{\vec{y}}$ is H-orthogonal to V. Such a projection is known to be the center of the set $N^{-1}(\vec{y}) \cap F_1$, and hence $q_{\vec{y}}$, being central, is an optimal algorithm; see e.g. Traub, Wasilkowski & Woźniakowski (1988, Chap. 4). □

Thus the optimal network uses a number n of neurons equal to the number k of examples, and the corresponding activation functions are centered at the information points $\vec{t}_j$. The optimal coefficients of the network are obtained as solutions of a $k \times k$ linear system.

Remark 2. Throughout this paper information is, for simplicity, assumed noiseless. Assume for a moment that information is in addition contaminated by noise uniformly bounded in some norm, i.e., $\vec{y} = Nf + \vec{\eta}$ with $\|\vec{\eta}\|_Y \le \delta$ ($\delta > 0$). Then an almost optimal network (i.e., optimal up to a factor of 2) is the one minimizing the regularization functional

$$\mathcal{F}(g) = (\delta^2/a^2)\|g\|_H^2 + \|\vec{y} - N(g)\|_Y^2$$

over $g \in H$. That is, the minimum of $\mathcal{F}$ is uniquely determined and is a network with k neurons. In particular, for $\|\cdot\|_Y$ a Hilbert norm, $\|\vec{\eta}\|_Y = \sqrt{\vec{\eta}^T \Sigma \vec{\eta}}$ with $\Sigma = \Sigma^T > 0$, this minimum is given as in Theorem 2

with the coefficients c_j being the solution of the linear system $((\delta^2/a^2)\Sigma + M_N)\vec{c} = \vec{y}$.

This is a direct consequence of results in analytic complexity given in Plaskota (1996, Sec. 2.5-6). We additionally note that for noise bounded in a Hilbert norm the coefficients c_j can be selected in such a way that the network $q_{\vec{y}}$ is optimal, i.e., it minimizes the error of approximation in the class F_1. This follows from results in Melkman & Micchelli (1979). However, an explicit construction of these optimal coefficients is in general unknown. The types of algorithms mentioned above are also studied in Girosi & Poggio (1990). It is interesting that these are best algorithms in the above strict sense.

Assume now the general case with arbitrary n and k. This, in particular, includes the case when the number n of available neurons is smaller than the number k of examples.

Theorem 3. *Let $f \mapsto Al(f)$ and $\vec{y} \mapsto \phi(\vec{y})$ be almost optimal algorithms for finding neural network approximations, respectively, for n neurons and full information about f, and for an arbitrary number of neurons and information $\vec{y} = Nf$ about f. That is,*

$$\|f - Al(f)\|_e \le C_1 \cdot e_1(n) \quad \textit{and} \quad \|f - \phi(Nf)\|_e \le C_2 \cdot r(N),$$

for all $f \in F_1$. Then the composite algorithm $Al \circ \phi$ gives an almost optimal network in $\mathcal{N}_n$, i.e.,

$$\|f - Al(\phi(Nf))\|_e \le (C_1 + C_2.) \cdot e(n, N), \qquad \forall f \in F_1.$$

Proof. By the triangle inequality, for the composite algorithm

$$\begin{aligned} \|f - Al(\phi(Nf))\|_e &\le \|f - \phi(Nf)\|_e + \|\phi(Nf) - Al(\phi(Nf))\|_e \\ &\le C_2 r(N) + C_1 e_1(n) \le (C_1 + C_2) \max\left(e_1(n), r(N)\right) \\ &\le (C_1 + C_2) e(n, N), \end{aligned}$$

as claimed. □

Let us comment on Theorem 3 above. It says that the problem of finding an (almost) optimal network can be divided onto two separate problems related to pure neural and information issues. In the first step, an algorithm ϕ using $n_1 = k$ neurons approximates the i-o function f. In the second this approximate

function $q_y = \phi(\vec{y})$ is used as a target function (now in the full information setting) for an approximation Al by the current $n < n_1$ neural network. Thus the composite algorithm $Al \circ \phi$ gives an almost optimal network.

We now pass to the complexity questions.

From the proof of the last theorem we can immediately infer the following result, allowing us to bound (up to a factor of 2) the error $e(n, N)$ in terms of $e_1(n)$ and $e_2(N)$, and the error $e(n, k)$ by $e_1(n)$ and $e_2(k)$.

Theorem 4. *(a) For $k \leq n$ we have*

$$e_1(n) \leq e(n, N) = e_2(N), \qquad card(N) = n,$$

$$e_1(n) \leq e_1(k) \leq e(n, k) = e_2(k).$$

(b) For arbitrary k and n we have

$$\max\,(e_1(n),\, e_2(N)\,) \leq e(n, N) \leq e_1(n) + e_2(N), \qquad card(N) = n,$$

$$\max\,(e_1(n),\, e_2(k)\,) \leq e(n, k) \leq e_1(n) + e_2(k).$$

Proof. The first inequalities in (a) are a consequence of the fact that for $k \leq n$ the optimal network is the center of the set $N^{-1}(\vec{y}) \cap F_1$, while the first inequalities in (b) can be obtained by applying the composite algorithm of the previous theorem with Al and ϕ such that their errors are arbitrarily close to $e_1(n)$ and $e_2(N)$, respectively. The second inequalities in (a) and (b) then follow from the first ones by taking the infima over N with $card(N) = k$. □

Corollary 1. *A necessary condition for the error $e(n, k)$ to be at most ϵ is that $e_1(n),\, e_2(k) < \epsilon$, while a sufficient condition is that $e_1(n),\, e_2(k) \leq \frac{1}{2}\,\epsilon$. In other words, in order to approximate an i-o function with error at most ϵ, we must use at least $k(\epsilon)$ examples of f and a network with $n(\epsilon)$ neurons, while this error of approximation can be obtained via an optimal algorithm using at most $k(\epsilon/2)$ examples and $n(\epsilon/2)$ neurons.*

Inverting the relationship $e_1(k) \leq e_2(k)$ in Theorem 4 and using the fact both functions are monotone decreasing, we obtain Theorems III and IV from Section 2.

We have showed in particular that $e_1(n) \le e_2(n)$. Sometimes, but not always, we also have the reverse inequality, i.e., $e_2(n) \le C\, e_1(n)$ for some constant C independent of n, which means that the both minimal errors are essentially the same; see Example 2. In this case, there is another algorithm for constructing approximation networks consisting of the two following steps. In the first step we find, as before, the network $q_{\vec{y}}$ with $n_1 = k > n$ neurons best approximating f from the given k examples. In the second step we find the network $q^*_{\vec{y}}$ best approximating $q_{\vec{y}}$ from n examples of $q_{\vec{y}}$. Specifically let $N^*(g) = (g(\vec{t}^{\,*}_1), \dots, g(\vec{t}^{\,*}_n))^T$ be optimal (or almost optimal) information, i.e., information for which $e(N^*) \le C_3\, e_2(n)$ for some C_3 independent of n. Letting $\vec{y}^{\,*} = N^*(q_{\vec{y}})$ and $M_{N^*} = (G(\vec{t}^{\,*}_i, \vec{t}^{\,*}_j))_{i,j=1}^n$, we find $q^*_{\vec{y}} = \sum_{i=1}^n c^*_j G(\vec{t}^{\,*}_j, \cdot\,)$ with $M_{N^*}\vec{c}^{\,*} = \vec{y}^{\,*}$. Thus we have made one-time use of a network with $n_1 > n$ neurons in order to construct the weights c^*_j of the final network having n neurons. Note that in the second step we use only partial information $N^*(q_{\vec{y}})$ about $q_{\vec{y}}$, though full information is available.

Using again the triangle inequality we get for any $f \in F_1$,

$$\begin{aligned}\|f - q^*_{\vec{y}}\| \le \|f - q_{\vec{y}}\| + \|q_{\vec{y}} - q^*_{\vec{y}}\| &\le C_2\, e_2(k) + C_3\, e_2(n) \\ &\le (C_2 + C_3 C)\max\,(e_1(n), e_2(k)),\end{aligned}$$

i.e., the new algorithm is also almost optimal.

We finally present an example illustrating the above results.

Example 2. Let $D = [0,1]$ and the activation functions be given by $G(s,t) = \min(s,t)$, as in Example 1. Recall that then any neural network in $\mathcal{N}_n$ is a continuous and piecewise linear function with n knots, and the space H consists of absolutely continuous real functions defined on D and vanishing at zero. We assume as earlier that our prior knowledge of the i-o function f is that $\|f\|_H = \sqrt{\int_0^1 (f'(t))^2 dt} \le a$ for a fixed a.

Suppose first that we wish to approximate f in L^2 norm, i.e., $\|f\|_{\mathrm{e}} = \sqrt{\int_0^1 f^2(x)dx}$. Then $e_2(n)$ and $e_1(n)$ are both proportional to $1/n$ and equidistant sampling is close to optimal. Indeed, to calculate $e_2(n)$ we can use the well-known formula for the radius of information (see, e.g., Traub, Wasilkowski, & Woźniakowski, 1988, Chap. 4),

$$r(N) = \sup\{\, \|f\|_e :\ \|f\|_H \le a,\ Nf = \vec{0}\,\}.$$

Since $e_1(n) \leq e_2(n)$, the error $e_1(n)$ is also at most proportional to $1/n$. The lower bound for $e_1(n)$, proportional to $1/n$, can be in turn obtained for the saw-tooth function which takes zeros at $2n$ equidistant knots. Hence, for L_2 approximation, the neural and information complexities are comparable and both proportional to $1/\epsilon$.

Consider now the uniform error norm $\|f\|_e = \sup_{0\leq x\leq 1}|f(x)|$. Using a similar argument as for L^2 norm, we get that equidistant sampling is again (almost) optimal. However, the error $e_2(n)$ is now proportional to $1/\sqrt{n}$, and the information complexity increases to $(1/\epsilon)^2$. This is not surprising as the uniform norm is stronger than the L^2 norm. It is interesting, however, that $e_1(n)$ remains proportional to $1/n$ (as for L^2 norm), and the neural complexity is proportional to $(1/\epsilon)$. To see this, for a given $f \in F_1$ we select the knots $0 = t_0 < t_1 < \cdots < t_n = 1$ such that for all i

$$(t_i - t_{i-1}) \int_{t_{i-1}}^{t_i} (f'(t))^2 dt \leq \frac{a^2}{n^2}.$$

Note that this is possible since $\int_0^1 (f'(t))^2 dt \leq a^2$. As the network approximating f we take the piecewise linear interpolation q_f of f with the selected knots t_i. Then for any $x \in [t_{i-1}, t_i]$ we have

$$\begin{aligned}
\Big|f(x) - q_f(x)\Big| &= \left|\int_{t_{i-1}}^{x} f'(t) - q_f'(t)\, dt\right| \\
&\leq \int_{t_{i-1}}^{x} |f'(t) - q_f'(t)|\, dt \\
&\leq \sqrt{(t_i - t_{i-1})}\sqrt{\int_{t_{i-1}}^{t_i} (f'(t) - q_f'(t))^2\, dt} \leq \frac{a}{n},
\end{aligned}$$

as claimed. The lower bound for $e_1(n)$ is again obtained for the saw-tooth function.

Thus neural complexity is an order of magnitude smaller than information complexity for error measured in the uniform norm, and both complexities are roughly the same for error measured in L^2 norm. This is a consequence of the fact that for the uniform norm an adaptive choice of knots (different knots for different f's) is better than a nonadaptive one, while this is not the case for L^2 norm.

References

Aronszain, N. (1950). Theory of reproducing kernels. Transactions of the American Mathematical Society, 68, 337-404.

Barron, A.R. (1993). Universal approximation bounds for superposition of a sigmoidal function. IEEE Transactions in Information Theory, 39, 930-945.

Chui, C.K., Li, X., & Mhaskar, H.N. (1993). Neural networks for localized approximation. Center for Approximation Theory Report, 289.

Chui, C.K., Li, X., & Mhaskar, H.N. (1996). Limitations of the approximation capabilities of neural networks with one hidden layer. Advances in Computational Mathematics, 5, 233-243.

Chui, C.K., & Li, X. (1992). Approximation by ridge functions and neural networks with one hidden layer. Journal of Approximation Theory, 70, 131-141.

Girosi, F., & Poggio, T. (1990). Networks and the best approximation property. Biological Cybernetics, 63, 169-176.

Melkman, A.A., & Micchelli, C.A. (1979). Optimal estimation of linear operators in Hilbert spaces from inaccurate data. SIAM Journal on Numerical Analysis, 16, 87-105.

Mhaskar, H.N. (1996). Neural Networks and Approximation Theory. Neural Networks, 9, 721-722.

Mhaskar H.N., & Micchelli, C.A. (1992). Approximation by superposition of sigmoidal and radial basis functions. Advances in Applied Mathematics, 13, 350-373.

Mhaskar H.N., & Micchelli, C.A. (1994). Dimension independent bounds on the degree of approximation by neural networks. IBM Journal of Research and Development, 38, 277-284.

Mhaskar H.N., & Micchelli, C.A. (1995). Degree of approximation by neural and translation networks with a single hidden layer. Advances in Applied Mathematics, 161, 151-183.

Mhaskar H.N., & Micchelli, C.A. (1993). How to choose an activation function. In S.J. Hanson, J.D. Cowan, & C.L. Giles (Eds.), Advances in Neural

Information Processing Systems 5. San Mateo, CA: Morgan Kaufmann Publishers.

Micchelli, C.A., & Buhmann, M. (1992). On radial basis approximation on periodic grids. Mathematical Proceedings of the Cambridge Philosophical Society, 112, 317-334.

Pinkus, A. (1985). n-Widths in Approximation Theory. Springer Verlag, Berlin.

Plaskota, L. (1996). Noisy Information and Computational Complexity. Cambridge University Press, Cambridge.

Poggio, T., & Girosi, F. (1990). Regularization algorithms for learning that are equivalent to multilayer networks. Science, 247, 978-982.

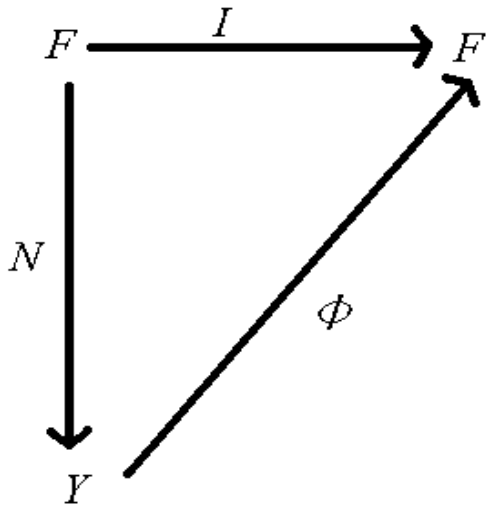
Poggio, T., & Girosi, F. (1989). A theory of networks for approximation and learning. Artificial Intelligence Memo., 1140, M.I.T. A.I. Lab.

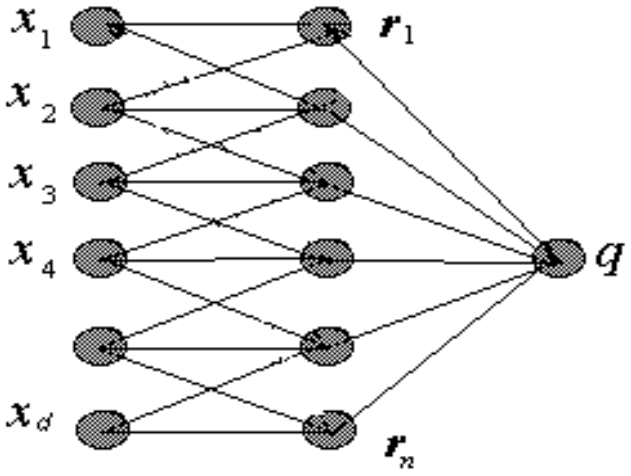
Traub, J.F., Wasilkowski, G.W., & Woźniakowski, H. (1988). Information-Based Complexity. Academic Press, Boston.

Figures:

Fig. 1: The schematic relation of the spaces in continuous complexity theory

Fig. 2: Model three layer network


F
I
F
N
ϕ
Y


x_1
x_2
x_3
x_4
x_d
r_1
r_n
q