CONF-960797--1

PAC Learning Algorithms for Functions Approximated by Feedforward Networks†

Nageswara S. V. Rao
Vladimir Protopopescu
Center for Engineering Systems Advanced Research
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831-6364

Submitted to: *13th International Conference on Machine Learning*, Bari, July 3-6, 1996.

†Research sponsored by the Engineering Research Program of the Office of Basic Energy Sciences, of the U.S. Department of Energy, under Contract No. DE-AC05-96OR22464 with Lockheed Martin Energy Research Corp.

PAC Learning Algorithms for Functions Approximated by Feedforward Networks†

Nageswara S. V. Rao and Vladimir Protopopescu
Center for Engineering Systems Advanced Research
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831-6364

Abstract

We present a class of efficient algorithms for PAC learning continuous functions and regressions that are approximated by feedforward networks. The algorithms are applicable to networks with unknown weights located only in the output layer and are obtained by utilizing the potential function methods of Aizerman *et al.* [1]. Conditions relating the sample sizes to the error bounds are derived using martingale-type inequalities. For concreteness, the discussion is presented in terms of neural networks, but the results are applicable to general feedforward networks, in particular to wavelet networks. The algorithms can be directly adapted to concept learning problems.

Keywords and Phrases: Potential functions, PAC learning, efficient learning algorithms.

---

†Research sponsored by the Engineering Research Program of the Office of Basic Energy Sciences, of the U.S. Department of Energy, under Contract No. DE-AC05-96OR22464 with Lockheed Martin Energy Research Corp.

1 Introduction

The problem of learning (or inferring) a function or a set (concept) from a finite set of examples has been the focus of considerable research in areas such as pattern recognition, machine learning, neural networks, etc. From a theoretical viewpoint, machine learning problems have been extensively studied in the past decade under the Probably and Approximately Correct (PAC) paradigm pioneered by Valiant [23]. The main attractive feature of this paradigm lies in the ability to quantify the learning performance based on *finite* samples. However, many results in this direction are only existential in nature. Indeed, they establish that a hypothesis consistent with a sample is a PAC approximator, but they do not specify algorithms to compute that required hypothesis. The difficulty of the problem is compounded by the fact that - due to the strict consistency requirement - most computational problems are NP-hard. From an application perspective, neural networks have been employed in a number of practical function estimation problems. One of the most appealing features of neural network methods is that the learning algorithms are very simple to implement. Yet the performance of these algorithms for learning based on *finite samples* has been assessed only in very limited cases [18].

In this paper, we present a class of algorithms that solve the function estimation problem for continuous functions that are approximated by a special class of feedforward networks. These algorithms are easily implementable and at the same time provide performance easily quantifiable guarantees based on finite samples.

General density results guarantee that a continuous function can be approximated by a finite-sized network of non-polynomial units with a single hidden layer within a specified precision [14], [15]. (When the units are sigmoid functions, the networks are called *feedforward artificial neural networks* [7], [10], [2].) Density properties of sigmoidal feedforward networks with two hidden layers have been studied by Kurkova [13] by using Kolmogorov's superposition theorem [12]. Similar density properties of slightly different architectures based on wavelet networks have been studied by Zhang and Benveniste [26]. These density results enable us to formulate the function learning problem as one of estimating *finite dimensional* vectors (that typically represent the connection weights of a network) at the cost of settling for an approximation. The learning methods that compute the connection weights of a required network based on a finite sample have been extensively studied recently both analytically and heuristically. The recent renewal of interest in such methods can be attributed, at least in part, to the success of neural networks in a wide variety of applications. Typically, the performance of such methods depends on (a) the form of network employed, and (b) the learning algorithm that computes the parameters of the network.

We illustrate the application of the potential function method (Aizerman *et al.* [1]) Benveniste *et al.* [3]) to obtain learning algorithms implemented on feedforward network architectures. The method is applicable to feedforward networks with unknown weights located only in the output layersuch as Kurkova's networks [13]. Yet our approach can also be used to obtain: (a) efficient algorithms for learning sets (concepts) - for which most existing methods are non-algorithmic (Valiant [23], Natarajan [16]), and (b) learning algorithms for wavelet networks [26] - for which no finite sample results are known and no existing learning algorithms are shown to converge.

For the task of learning functions from a *finite sample*, the utility of the above density

results depends critically on the availability of suitable learning (or training) algorithms. There are several algorithms that train the networks of sigmoidal units based on finite samples (Werbos [25], van der Smagt [24], Tang and Koehler [22]). The performance of such algorithms has been assessed only to a limited extent and mostly in asymptotic cases. The popular backpropagation algorithm (Werbos [25], Rumelhart et al. [20]), which is a gradient descent method based on mean square error, seems to be effective in some cases but very slow to converge in others. Significant effort has been spent to improve the convergence of this and similar gradient search algorithms (Darken and Moody [8], Jacobs [11], Saarinen et al. [21], Chen and Lai [6], Fitch et al. [9]). Convergence properties of learning algorithms based on wavelet networks [26] are unknown. Also to our knowledge, no learning algorithms have been published for the networks based on Kurkova's model [13].

The organization of the paper is as follows. Some preliminary discussion on function and regression learning problems, neural network approximations, and potential function algorithms, are presented in Section 2. The potential function algorithms are utilized in conjunction with Kurkova's networks [13] to learn arbitrary continuous functions and regressions in Section 3. The concept learning problems and wavelet network algorithms are briefly described in Section 4.

2 Preliminaries

We first provide basic formulations of the function and concept learning problems. We then discuss the required density properties of networks proposed by Cybenko [7] and Kurkova [13]. Then we briefly summarize the potential function method developed by Aizerman *et al.* [1]. Throughout the paper, X and x denote random and deterministic variables, respectively, and it is assumed that all the functions satisfy the required measurability conditions.

2.1 Function and Regression Learning Problems

A *training n-sample* of a function $f : [0,1]^d \mapsto \Re$ is given by $(X_1, f(X_1))$, $(X_2, f(X_2))$, ..., $(X_n, f(X_n))$ where $X_1, X_2, \ldots, X_n$, $X_i \in [0,1]^d$, are independently and identically generated (iid) according to a distribution P_X ($X = [0,1]^d$). The *function learning problem* is to estimate a function $\hat{f} : [0,1]^d \mapsto \Re$, based on the sample, such that $\hat{f}(x)$ "closely" approximates $f(x)$. More precisely, we consider either the expected square error

$$I(\hat{f}) = \int [\hat{f}(X) - f(X)]^2 dP_X \qquad (2.1.1a)$$

or the expected absolute error

$$J(\hat{f}) = \int |\hat{f}(X) - f(X))| dP_X \qquad (2.1.1b)$$

which is to be minimized over a family of functions $\mathcal{F}$ based on the given n-sample. Let $f_* \in \mathcal{F}$ minimize $I(\hat{f})$ (or $J(\hat{f})$) over all $\hat{f} \in \mathcal{F}$. In general, f_* cannot be computed from (2.1.1a) or (2.1.1b) since P_X is unknown. Furthermore, since no restrictions are placed on the underlying distribution, it will not always be possible to infer f_* (with probability one) based on a finite sample. Consequently, most often only an approximation $\hat{f}$ to f_* is feasible. We

shall provide (sufficient) conditions under which an approximation $\hat{f}$ to f_* can be computed such that for a sufficiently large sample we have

$$P[I(\hat{f}) - I(f_*) > \epsilon] < \delta \tag{2.1.2a}$$

or

$$P[J(\hat{f}) - J(f_*) > \epsilon] < \delta \tag{2.1.2b}$$

corresponding to (2.1.1a) and (2.1.1b) respectively for arbitrarily specified $\epsilon > 0$ and δ, $0 < \delta < 1$, where $P = P_X^n$ is the product measure on the set of all iid n-samples. Thus the "error" due to $\hat{f}$ is to be bounded within an arbitrarily specified precision ϵ of minimum possible error, with an arbitrarily specified confidence $1-\delta$ (given a sufficiently large sample).

In the same context, consider the more general *regression learning problem.* We are given a training sample (X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n) iid according to a probability distribution $P_{X,Y}$ ($X = [0,1]^d$,$Y = \Re$) such that $f(x) = E(Y|x)$. The problem is to compute an estimate $\hat{f}$ of the regression function f that satisfies the condition (2.1.2a) or (2.1.2b) with $P = P_{X,Y}^n$.

2.2 Function Approximation by Neural Networks

Consider feedforward networks with two hidden layers that have been studied by Kurkova [13]. It can be shown (Theorem 2 of [13]) that any continuous function can be represented within an arbitrarily specified precision ϵ in the following form:

$$\sum_{q=1}^{m} \left[\sum_{j=1}^{m(m+1)^n} \left(d_j \rho \left(\sum_{p=1}^{n} \sum_{i=1}^{m+1} v_j w_{pq} a_{qi} \rho(b_{qi} x_p + c_{qi}) \right) + u_j \right) \right]$$

where $\rho(.)$ is a fixed function and the other parameters depend on the function to be learned. Furthermore, this function can be represented in the following simpler algebraic form

$$\sum_{i=1}^{M} a_i \eta_i(x) \tag{2.2.1}$$

where the functions $\eta_i(.)$ are universal and the weights a_i's depend on the function being approximated. The functions $\eta_i(.)$ correspond to single hidden layer feedforward networks consisting of sigmoid functions (see Kurkova [13] for details on the construction of these functions). As shown in the original formulation of Kolmogorov [12], when $\epsilon = 0$, the elemental functions η_i are highly non-smooth functions, which do not seem to be directly amenable to computer implementations.

2.3 Potential Function Method

The potential function algorithm was proposed by Aizerman *et al.* [1]. Consider a function of the form

$$f(x) = \sum_{i=1}^{M} a_i \phi_i(x) \tag{2.3.1}$$

where $\phi_i(x)$ form a linearly independent set of functions. Now for some real $\lambda_1, \lambda_2, \ldots, \lambda_M$ let

$$K(y,z) = \sum_{i=1}^{M} \lambda_i^2 \phi_i(y)\phi_i(z). \tag{2.3.2}$$

Given the sequence $(X_1, f(X_1)), (X_2, f(X_2)), \ldots$ consider the following algorithm

$$f^n(x) = f^{n-1}(x) + \frac{1}{\Lambda}[f(X_n) - f^{n-1}(X_n)]K(x, X_n) \tag{2.3.3}$$

such that $\Lambda > \frac{1}{2}\max_{y \in Y} K(y,y)$. The conditions under which $f^n(.)$ converges to $f(.)$ have been studied extensively. A survey of these results is provided in [1]; our application involves the results shown by Braverman and Pjatnickii [5], which deal with the case where M is finite. Since in Kurkova's networks [13] only the weights of the output layer depend on the function to be determined, we can apply the potential function method to wide classes of functions. On the contrary, in Cybenko's networks [7] the weights of both hidden and output layers depend on the function to be approximated. Hence these networks are not directly amenable to potential function methods, but they can be handled by alternative methods such as the stochastic approximation [17]).

3 Learning Algorithms Based on Potential Functions

Given a finite sample $(X_1, f(X_1)), (X_2, f(X_2)), \ldots, (X_n, f(X_n))$, consider the algorithm (2.3.3) which can be implemented in terms of coefficients as follows

$$a_i^n = a_i^{n-1} + \frac{1}{\Lambda}[f(X_n) - f^{n-1}(X_n)]\phi_i(X_n). \tag{3.1}$$

We will now provide sufficient conditions under which algorithms of this type can be used for solving the function and regression learning problems.

3.1 Exactly Representable Functions

The following condition is utilized for the potential function method.

Condition 3.1 *For a fixed M, any function $f \in \mathcal{F}$ is given by $f(x) = \sum_{j=1}^{M} a_j \phi_j(x)$, where w is the parameter vector with components a_i such that $\sum_{j=1}^{M} a_j^2 \neq 0$, and $\int_X f^2(x) dP_X > 0$.*

This condition is satisfied if $f(.)$ is continuous and vanishes at no more than a finite number of points. This condition implies that the $M \times M$ matrix $[\rho_{ij}] = [\int_X \phi_i(x)\phi_j(x)p(x)dx]$ is positive definite. Thus

$$r\sum_{i=1}^{M} a_i^2 \leq \sum_{i=1}^{M}\sum_{j=1}^{M} a_i \rho_{ij} a_j \leq R\sum_{i=1}^{M} a_i^2$$

where r and R are the smallest and largest eigenvalues of the matrix $[\rho_{ij}]$.

Theorem 3.1 *Suppose the sample size, n, is given by*

$$n = \ln(\delta\epsilon/RC) / \ln(1 - ra)$$

where $C = \sum_{i=1}^{M} a_i^2$ *and* $a = \frac{1}{\Lambda}\left[2 - \frac{\max_{x \in X} K(x,x)}{\Lambda}\right]$, *with* $1 - ra \geq 0$, *where* r *and* R *are the smallest and largest eigenvalues of the matrix* $[\rho_{ij}]$, *and* Λ *is a free parameter chosen such that* $a > 0$. *Then under Condition 3.1, for* $f \in \mathcal{F}$ *and* f^n *produced by the algorithm (3.1), we have*

$$P[I(f^n) < \epsilon] > 1 - \delta.$$

Furthermore $I(f^n)$ *converges to* 0 *with probability one.*

Proof: The outline of the proof is direct: Braverman and Pyatnitskii (Theorem 1 of [5]) showed that $E[I(f^n)] \leq RC(1 - ra)^n$, which is combined with the Chebyshev's inequality to show the theorem.

We provide the details here for completeness (this proof can be found in [5] which makes use of results from earlier publications) and also to facilitate the proof of Theorem 3.4. Define the following quantities:

$$f(x) - f^n(x) = \sum_{i=1}^{M} \Delta a_i^n \phi_i(x)$$

where $\Delta a_i^n = a_i - a_i^n$ and $f(x) = \sum_{i=1}^{M} a_i \phi_i(x)$, and

$$\alpha_n = \sum_{i=1}^{M} (\Delta a_i^n)^2$$

$$I(f^n) = \beta_n = \int_X [f(X) - f^n(X)]^2 dP_X = \sum_{i=1}^{M} \sum_{j=1}^{M} \Delta a_i^n \rho_{ij} \Delta a_j^n.$$

We express α_{n+1} in terms of α_n as

$$\begin{aligned} \alpha_{n+1} &= \sum_{i=1}^{M} (\Delta a_i^n)^2 - 2r_{n+1}(\Delta a^n, \phi(X_{n+1})) + r_{n+1}^2 \sum_{i=1}^{M} [\phi_i(X_{n+1})]^2 \\ &= \alpha_n - 2r_{n+1}[f(X_{n+1}) - f^n(X_{n+1})] + r_{n+1}^2 K(X_{n+1}, X_{n+1}) \end{aligned}$$

where $\phi(x) = (\phi_1(x), \phi_2(x), \ldots, \phi_M(x))^T$, $\Delta a^n = (a_1 - a_1^n, a_2 - a_2^n, \ldots, a_M - a_M^n(x))^T$ and $r_n = \frac{1}{\Lambda}[f(X_{n+1}) - f^n(X_{n+1})]$. By taking conditional expectations on both sides of the above equation we obtain

$$\begin{aligned} E[\alpha_{n+1} | \Delta a^n] &= \alpha_n - \frac{2}{\Lambda} \int_X [f(X) - f^n(X)]^2 dP_X + \frac{1}{\Lambda^2} \int_X K(X,X)[f(X) - f^n(X)]^2 dP_X \\ &\leq \alpha_n - a \int_X [f(X) - f^n(X)]^2 dP_X \end{aligned}$$

where $a = \frac{2}{\Lambda} - \frac{Q}{\Lambda^2}$ and $\max_x K(x,x) \leq Q$. In summary we have

$$E[\alpha_{n+1}|\Delta a^n] \leq \alpha_n - a\beta_n.$$

By taking expectations on both sides we obtain

$$E[\alpha_{n+1}] \leq E[\alpha_n] - aE[\beta_n].$$

Since the matrix $[\rho_{ij}]$ is positive definite, we have the condition $r\alpha_n \leq \beta_n \leq R\alpha_n$, which yields the following inequality

$$rE[\alpha_n] \leq E[\beta_n] \leq RE[\alpha_n]$$

and

$$E[\alpha_n] \leq E[\alpha_{n-1}](1 - ra) \leq E[\alpha_0](1 - ra)^n.$$

Then by taking $f^0(.) = 0$, we have $E[\alpha_0] = \sum_{i=1}^{M} a_i^2$, which yields $E[\beta_n] \leq RC(1 - ra)^n$ for sample size n. By Chebyshev's inequality we have

$$P[\beta_n > \epsilon] \leq \frac{E[\beta_n]}{\epsilon}.$$

The right hand side is equated to δ to obtain the bound on the sample size. The proof for almost sure convergence follows along the lines of [5]. □

The algorithm (3.1) uses constant the step-size $\frac{1}{\Lambda}$, which is referred to as the learning rate in the context of neural network algorithms. We now consider the case where the step-size is variable in the the following simpler algorithm:

$$f^n(x) = f^{n-1}(x) + \gamma_n sign[f(X_n) - f^{n-1}(X_n)]K(x, X_n) \tag{3.2}$$

where γ_n is a sequence of positive numbers such that $\sum_{i=1}^{\infty} \gamma_i \to \infty$ and $\sum_{i=1}^{\infty} \gamma_i^2 < \infty$. These are the well-known Robbins-Monro [19] conditions on the step-size; for example, the choice $\gamma_i = \gamma/(i+1)$, $\gamma > 0$ satisfies these conditions.

Theorem 3.2 *Suppose there exist a constant λ_0 and a sufficiently large sample size, n, such that $\delta\epsilon = \sqrt{RC}\gamma_n^{\lambda_0/2}$, where $C = \sum_{i=1} a_i^2$ and R is the largest eigenvalue of the matrix $[\rho_{ij}]$. Then under Condition 3.1, for $f \in \mathcal{F}$ and f^n produced by algorithm (3.2), we have*

$$P[I(f^n) < \epsilon] > 1 - \delta.$$

Moreover $I(f^n)$ converges to 0 with probability one.

Proof: The outline of the proof consists of first establishing that

$$E[\alpha_{n+1}] \leq E[\alpha_n] - 2\gamma_{n+1}E[\beta_n] + Q\gamma_{n+1}^2$$

and then showing that $E[\beta_n] \leq \sqrt{RC}\gamma_n^{\lambda_0/2}$. The details of the proof are like in Theorem 3.1 for the estimation of the sample size; Lemma A.1 of [17] is utilized to show the boundedness of expectations. Almost sure convergence results directly follow from [5]. □

Consider a variable step-size version of the algorithm (3.1) as follows:

$$f^n(x) = f^{n-1}(x) + \gamma_n[f(X_n) - f^{n-1}(X_n)]K(x, X_n) \tag{3.3}$$

where $\{\gamma_i\}$ is a sequence of positive numbers such that $\sum_{i=1}^{\infty} \gamma_i \to \infty$ and $\sum_{i=1}^{\infty} \gamma_i^2 < \infty$. We can state:

Theorem 3.3 *Suppose there exists a constant λ and a sufficiently large sample size, n, such that $\gamma_n = \left[\frac{\delta\epsilon}{RC}\right]^{1/\lambda}$, where $C = \sum_{i=1} a_i^2$ and R is the largest eigenvalue of the matrix $[\rho_{ij}]$. Then under Condition 3.1, for $f \in \mathcal{F}$ and f^n produced by the algorithm (3.3), we have*

$$P[I(f_n) < \epsilon] > 1 - \delta.$$

Moreover $I(f^n)$ converges to 0 with probability one.

Proof: The outline of the proof consists of first establishing that

$$E[\alpha_{n+1}] \leq E[\alpha_n] - \gamma_{n+1}E[\beta_n] + Q\gamma_{n+1}^2$$

and then showing that $E[\beta_n] \leq RC\gamma_n^{\lambda}$. The details of the proof are as in Theorem 3.1 with an application of Lemma A.1 of [17]. □

3.2 Approximately Representable Functions

We now address the question of using the algorithm of the form (2.3.3), based on functions of the form $f(x) = \sum_{i=1}^{M} a_i\phi_i(x)$, to approximate a function of the form $g(x) = \sum_{i=1}^{M_1} c_i\chi_i(x)$. For two functions f_1 and f_2, we define the distance functional

$$I(f_1, f_2) = \int_X [f_1(X) - f_2(X)]^2 dP_X.$$

Given an infinite sample $(X_1, g(X_1))$, $(X_2, g(X_2))$, ..., consider the algorithm (2.3.3) in the form

$$f^n(x) = f^{n-1}(x) + \frac{1}{\Lambda}[g(X_n) - f^{n-1}(X_n)]K_1(x, X_n). \tag{3.4}$$

Specifically, we are interested in the conditions under which $\{f^n\}$ converges to some f^*, and if f^* exists how good it is as an approximation to $g(x)$. The first part of the question is answered affirmatively in the next theorem. To answer the second part consider the best approximation g^* to g from among $\mathcal{F}$ in the following sense:

$$\epsilon^* = \min_{\hat{f} \in \mathcal{F}} I(\hat{f}, g)$$

$$\epsilon^* = I(f^*, g).$$

Then the following theorem shows that the error due to f^* can be made arbitrarily close to ϵ^* with arbitrarily high probability by using a sample of suitable size.

Theorem 3.4 *Suppose the sample size, n, is given by:*

$$n = \ln(\delta\epsilon/RC)\,/\ln(1-ra)$$

where $C = \sum_{i=1}^{M} a_i^2$, $a = \frac{1}{\Lambda}\left[2 - \frac{\max_{x\in X} K(x,x)}{\Lambda}\right]$, *with* $1 - ra \geq 0$, *r and R are the smallest and largest eigenvalues of the matrix $[\rho_{ij}]$, and Λ is a free parameter chosen such that $a > 0$. Then under Condition 3.1, for $g(x)$ of the form $g(x) = \sum_{i=1}^{M_1} c_i\chi_i(x)$, and f^n produced by the algorithm (3.4), we have*

$$P[I(f_n, g) < (\sqrt{\epsilon^*} + \sqrt{\epsilon})^2] > 1 - \delta.$$

Moreover, $I(f_n, g)$ converges to ϵ^* with probability one.

Proof: Consider $g(x) = \sum_{i=1}^{M_1} c_i\chi_i(x)$ and $f^*(x) = \sum_{i=1}^{M_1} a_i\phi_i(x)$. Without loss of generality assume that $M_1 = M$, since otherwise the required number of a_i's or c_i's with zero values can be added. We have

$$g(x) - f^n(x) = [g(x) - f^*(x)] + [f^*(x) - f^n(x)] = g(x) - f^*(x) + \sum_{i=1}^{M}(a_i - a_i^n)\phi_i(x).$$

As in Theorem 3.1 define $\alpha_n = \sum_{i=1}^{M}(a_i - a_i^n)^2$ and $\Delta a_i^n = a_i - a_i^n$. Then consider

$$\begin{aligned}
[g(x) - f^n(x)]^2 &= \left[\sum_{i=1}^{M} \Delta a_i^n \phi_i(x) + g(x) - f^*(x)\right]^2 \\
&= \left[\sum_{i=1}^{M}\sum_{j=1}^{M} \Delta a_i^n \phi_i(x)\phi_j(x)\Delta a_j^n\right]^2 + [g(x) - f^*(x)]^2 \\
&\quad +2[g(x) - f^*(x)]\sum_{i=1}^{M} \Delta a_i^n \phi_i(x).
\end{aligned}$$

By taking expectations on both sides we obtain

$$I(f^n, g) = \beta_n + \epsilon^* + 2\int_X [g(X) - f^*(X)]\sum_{i=1}^{M} \Delta a_i^n \phi_i(X) dP_X$$

where $\beta_n = \int_X [f^*(X) - f^n(X)]^2 dP_X$ and $\epsilon^* = I(f^*, g)$. By comparing with the proof of Theorem 3.1, we have same forms for α_n and β_n. Thus for the sample size stated here, with probability at least $1 - \delta$ we have

$$\int_X [f^*(X) - f^n(X)]^2 dP_X < \epsilon$$

which implies $|f^*(x) - f^n(x)| \leq \sqrt{\epsilon}$. Further $|g(x) - f^*(x)| \leq \sqrt{\epsilon^*}$, and hence the third term of the above equation is bounded above by $2\sqrt{\epsilon^*}\sqrt{\epsilon}$. Thus with probability at least $1 - \delta$ we have $I(f^n, g) \leq (\sqrt{\epsilon^*} + \sqrt{\epsilon})^2$. □

This theorem is useful in solving the general regression estimation problem. Also, variants of the theorem along the lines of Theorem 3.2. and 3.3 are straightforward.

3.3 Regression Estimation

We consider now the problem of minimizing

$$Q(\hat{f}) = \int_{X,Y} [\hat{f}(X) - Y]^2 dP_{X,Y}$$

over all $\hat{f} \in \mathcal{F}$ based on a finite sample. The regression function $f(x)$ minimizes $Q(\hat{f})$ since

$$Q(\hat{f}) = \int_{X,Y} [f(X) - Y]^2 dP_{X,Y} + I(\hat{f})$$

and $I(f) = 0$. By treating the sample as if it had been generated by a function f_{emp} (i.e. $y_i = f_{emp}(x_i)$ for $i = 1, 2, \ldots, n$), we compute an approximation $\hat{f}_{emp}$ to f_{emp} such that

$$P[I(\hat{f}_{emp}) > \epsilon_{emp}] < \delta$$

by using the potential function method. Then we show that $\hat{f}_{emp}$ is close to the regression function $f(x)$ in the following theorems under Condition 3.1. To illustrate the main point, we first solve the problem under the additional constraint that the sample is consistent with some function in $\mathcal{F}$; a more general form is provided later.

Theorem 3.5 *Suppose that Condition 3.1 is satisfied and the sample size, n, is sufficiently large, such that $\epsilon_{emp} = \frac{c}{\delta}(1 - ra)^n$, where c and r are constants. Moreover, suppose that $sup_{x,y,f}(y - f(x))^2 \leq \tau$ and $\mathcal{F}$ has finite capacity, such that $(X_1, Y_1), \ldots, (X_n, Y_n)$ is consistent with a function of $\mathcal{F}$. Then for $\delta = 9\, e^{-\epsilon^2 n/16\tau^2}$ and $\hat{f}(.) = f^n$ given by algorithm (3.1), we have*

$$P[I(\hat{f}) < (\sqrt{\epsilon} + \sqrt{\epsilon_{emp}})^2] > 1 - \delta.$$

Proof: Under Condition 3.1, the regression function $f(x) \in \mathcal{F}$ minimizes $Q(\hat{f})$. By the hypothesis that the sample is consistent with some function $f_{emp} \in \mathcal{F}$ we obtain that $Q_{emp}(f_{emp}) = 0$. By using the algorithm of last section, we obtain a approximation $\hat{f}_{emp}$ to f_{emp} such that for n given by $\epsilon_{emp} = \frac{c}{\delta}(1 - ra)^n$, we have

$$P[I(\hat{f}_{emp}, f_{emp}) > \epsilon_{emp}] < \delta.$$

Since f_{emp} minimizes empirical error Q_{emp} overall $\hat{f} \in \mathcal{F}$, from Theorem 2.1 of [17], for $\delta = 9\, e^{-\epsilon^2 n/16\tau^2}$ we have

$$P[|Q(f_{emp}) - Q(f)| > \epsilon] < \delta.$$

This condition is equivalent to

$$P[I(f_{emp}) > \epsilon] < \delta$$

since $Q(\hat{f}) - Q(f) = I(\hat{f})$. Then, we have

$$\begin{aligned} I(\hat{f}_{emp}) &= \int_X [f(X) - \hat{f}_{emp}(X)]^2 dP_X \\ &= \int_X [f_{emp}(X) - \hat{f}_{emp}(X)]^2 dP_X + \int_X [f_{emp}(X) - f(X)]^2 dP_X \\ &\quad + 2\int_X [f_{emp}(X) - \hat{f}_{emp}(X)][f(X) - f_{emp}(X)] dP_X \end{aligned}$$

where the first and second terms of right hand side are $I(\hat{f}_{emp}, f_{emp})$ and $I(f_{emp})$ respectively; with probability of at least $1-\delta$, these two quantities are bounded above by ϵ_{emp} and ϵ, respectively. Then the last term of the above equation is bounded above by $2\sqrt{\epsilon_{emp}}\sqrt{\epsilon}$. □

We now state a more general result applicable to the case when there is no $\hat{f} \in \mathcal{F}$ consistent with the sample; the proof follows along the lines of Theorems 3.1 and 3.5.

Theorem 3.6 *Let* $f^* = \min_{\hat{f}\in\mathcal{F}} I(\hat{f}, f)$ *and* $\epsilon^* = I(f^*, f)$. *Suppose the sample size,* n, *is sufficiently large, such that* $\epsilon_{emp} = \frac{c}{\delta}(1-ra)^n$, *where* c *and* r *are constants. Then under Condition 3.1 together with* $sup_{x,y,f}(y-f(x))^2 \leq \tau$ *and finite capacity of* $\mathcal{F}$, *we have*

$$P[I(\hat{f}, f) < (\sqrt{\epsilon^*} + \sqrt{\epsilon} + \sqrt{\epsilon_{emp}})^2] > 1-\delta$$

where $\delta = 9\, e^{-\epsilon^2 n/16\tau^2}$ *and* $\hat{f}(.) = f^n$ *is given by algorithm (3.1).*

4 Discussion

We have presented a class of convergent function learning algorithms implemented on neural networks. Two immediate generalizations are in order. First, we extend the learning paradigm to concepts where the problem is to learn indicator functions that are not necessarily continuous. Second, we extend the applicability of the learning algorithms to other classes of networks in particular to wavelet networks. Notice that the proof methods used in last two sections (with the exception of Section 3.1) depend critically on the architecture of the networks, but are independent of the particular nonlinearity used in the network. Thus these methods are applicable to more general feedforward networks of suitable structure. We illustrate this aspect by applying the learning algorithms to the wavelet networks.

4.1 Concept Learning

We consider now the framework of *concept learning* proposed by Valiant [23]. We are given a set $\mathcal{X} = [0,1]^d$ called the *domain*, and $C \subseteq 2^{\mathcal{X}}$and $H \subseteq 2^{\mathcal{X}}$ called the *concept class* and *hypothesis class* respectively; members of C and H are measurable under a distribution P_X on $\mathcal{X}$. A *concept* is any $c \in C$ and a *hypothesis* is any $h \in H$. For $s \subseteq \mathcal{X}$, an *indicator function* $1_s : \mathcal{X} \mapsto \{0,1\}$ is defined such that for any $x \in \mathcal{X}$, $1_s(x) = 1$ (0) if and only if $x \in s$ $(x \notin s)$. A pair $(x, 1_c(x))$ is called an *example of* $c \in C$, and set of m such examples is called *m-sample of c*. For $a, b \subseteq \mathcal{X}$, we have $a\Delta b = (\bar{a} \cap b) \cup (a \cap \bar{b})$.

The concept class C is said to be *learnable* [4] if given a finite sample, a hypothesis $h \in H$, $C \subseteq H$, can be produced such that for any $0 < \epsilon, \delta < 1$, we have the following condition satisfied

$$P[\mu(h\Delta c) \leq \epsilon] \geq 1-\delta. \tag{4.1}$$

Note that $\mu(h\Delta c)$ is the probability that a randomly chosen test point $X \in \mathcal{X}$ will be classified differently by h and c, i.e. $\mu(h\Delta c) = \int_{I_c(X)\neq I_h(X)} dP_X$.

The connection of PAC learning to the network learning is twofold: first, PAC learning provides a framework to obtain finite sample results for the network learning problem, and

second, the network learning provides constructive algorithms to solve several PAC learning classes (many results in PAC learning do not directly yield algorithmic solutions).

Let f be an indicator function (which is not necessarily a continuous function) of any finite measurable set of $[0,1]^d$. For any $\varepsilon > 0$, for some M, there exists [7] a function $g(x)$ of the form (2.2.1) and a set $D \subset [0,1]^d$ with Lesbegue measure of at least $1-\varepsilon$ and $|g(x) - f(x)| < \varepsilon$ for $x \in D$. We impose a condition analogous to Condition 3.1 as follows:

Condition 4.1 *For $\mathcal{X} = [0,1]^d$ and for a fixed M, the set of functions of the form*

$$h(w,x) = \sum_{j=1}^{M} a_j \sigma(b_j^T x - t_j)$$

approximates the set of indicator functions $\{1_c(.)\}_{c \in C}$ of the concept class C in the sense that for each $1_c(.)$ and $\varepsilon > 0$, there exists some $h(w,.)$ such that $|1_c(X) - h(w,X)| < \varepsilon$ for all $X \in D \subseteq \mathcal{X}$ where the Lebesgue measure of D is at least $1-\varepsilon$.

By using Kurkova's results [13], density of functions of the form (2.2.2) in the set of indicator functions (as in Condition 4.1) can be established along the lines of [7]. Thus the potential function method of Section 3 can be employed to solve the concept learning problem; in this formulation note that $I[h(w,.)] = \int [h(w,X) - 1_c(X)]^2 dP_X = \mu(h(w,.) \Delta c)$. Also, the regression estimation methods of Section 3 can be used to handle the cases where the membership functions are probabilistically defined. Finally, the case where the indicator functions of the concepts are not exactly represented as in Condition 4.1, results along the lines of Theorem 3.4 can be derived for the PAC concept learning problem.

4.2 Wavelet Networks

Consider the *wavelet network* of Zhang and Benveniste [26] of the form

$$h(w,x) = \sum_{i=1}^{M} a_i \psi(D_i x - t_i) + \bar{g} \tag{4.2}$$

where $a_i \in \Re$, $\psi : \Re^d \mapsto \Re$ is a *wavelet function* $t_i \in \Re^d$, $g \in \Re$, and D_i is $d \times d$ diagonal matrix with the diagonal entries given by $d_i \in \Re^d$. Let $\psi_s : \Re \mapsto \Re$ be a *scalar* wavelet in the Morlet-Grossmann sense [26] in that its Fourier transform $\widehat{\psi_s}(\omega)$ satisfies the condition

$$C_{\psi_s} = \int_0^{+\infty} \frac{|\widehat{\psi_s}(\omega)|^2}{\omega} d\omega \quad < \quad \infty.$$

Then the desired wavelet can be obtained by $\psi(x) = (\psi_s(x_1) \ldots \psi(x_d))$ where $x = (x_1, \ldots, x_d)$. The general architecture of a wavelet network is given in (4.2). Let w denote the *parameter vector* of the network which consists of a_1, a_2, ..., a_M, $d_1, d_2, \ldots, d_M$, $\bar{g}$ and $t_1, t_1, \ldots, t_M$. The wavelet networks satisfy density properties that are analogous to those in (2.3.1). Thus the wavelet networks can also be used for concept learning by using the following result.

Theorem 4.1 *Let f be an indicator function of any finite measurable set of $[0,1]^d$. For any $\varepsilon > 0$, for some positive integer M, there exists*

$$g(x) = h(w,x) = \sum_{i=1}^{M} a_i \psi(D_i x - t_i) + \bar{g} \tag{4.3}$$

and a set $D \subset [0,1]^d$ with Lebesgue measure of at least $1-\varepsilon$ such that $|g(x) - f(x)| < \varepsilon$ for $x \in D$.

Proof: Sums of the form (5.3) are shown in [26] to be dense in $L^2(\Re^d)$. The theorem follows by noting that the indicator functions on $[0,1]^d$ belong to $L^2(\Re^d)$ (see [7] for detailed proof for the networks of sigmoid functions; this proof can be adapted to the present case). □

Finally, wavelets can be employed to play a role similar to the sigmoids in the networks proposed by Kurkova [13]; in particular, the η_i's in (2.2.2) can be obtained by suitably using wavelet based functions. In this case the wavelet networks are also amenable to the potential function methods of Section 3.1.

References

[1] M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer. *Extrapolative problems in automatic control and method of potential functions*, volume 87 of *American Mathematical Society Translations*, pages 281–303. 1970.

[2] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):931–945, 1993.

[3] A. Benveniste, M. Metivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations*. Springer-Verlag, New York, 1990.

[4] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association of Computing Machinery*, 36(4):929–965, 1989.

[5] E. M. Braverman and E. S. Pjatnickii. Estimation of the rate of convergence of algorithm based on the potential functions method. *Automation and Remote Control*, 27(1):80–100, 1966.

[6] C. H. Chen and H. Lai. A comparison study of the gradient descent and the conjugate gradient backpropagation neural networks. In *Proceedings of the World Congress on Neural Networks*, volume III, pages 401–406, 1993.

[7] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Controls, Signals, and Systems*, 2:303–314, 1989.

[8] C. Darken and J. Moody. Towards faster stochastic gradient search. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances In Neural Information Processing Systems*, volume 4, pages 1010–1016. Morgan Kaufmann Publishers, San Mateo, California, 1992.

[9] J. P. Fitch, S. K. Lehman, F. U. Dowla, S. Y. Lu, E. M. Johansson, and D. M. Goodman. Ship wake detection procedure using conjugate gradient training artificial neural networks. *IEEE Trans. on Geoscience and Remote Sensing*, 29(5):718–726, 1991.

[10] K. Funahashi. On the approximate realizations of continuous mappings by neural networks. *Neural Networks*, 2:183–192, 1989.

[11] R. A. Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1:295–307, 1988.

[12] A. Kolmogorov. On the representation of continuous functions of many variables by superpositions of continuous functions of one variable and addition. *Doklady Akademii Nauk SSSR*, 114(5):953–956, 1957.

[13] V. Kurkova. Kolmogorov's theorem and multilayer neural networks. *Neural Networks*, 5:501–506, 1992.

[14] M. Leshno, V. Ya. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6:861–867, 1993.

[15] H. N. Mhaskar and C. A. Micchelli. Approximation by superposition of sigmoidal and radial basis functions. *Advances in Applied Mathematics*, 13:350–373, 1992.

[16] B. K. Natarajan. *Machine Learning: A Theoretical Approach.* Morgan Kaufmann Pub. Inc., San Mateo, California, 1991.

[17] N. S. V. Rao, V. Protopopescu, R. C. Mann, E. M. Oblow, and S. S. Iyengar. Learning algorithms for feedforward networks based on finite samples. Technical Report ORNL/TM-12819, Oak Ridge National Laboratory, September 1994. to appear in IEEE Trans. on Neural Networks.

[18] N. S. V. Rao, V. R. R. Uppuluri, and E. M. Oblow. Stochastic approximation algorithms for classes of PAC learning problem. In *Proceedings of 1994 IEEE Conf. on Neural Networks*, pages 791–796. 1994. a complete version to appear in IEEE Trans. on Systems, Man, and Cybernetics.

[19] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel and Distributed Processing: Explorations in the Microstructures of Cognition.* MIT Press, Cambridge, MA, 1986.

[21] S. Saarinen, R. B. Bramley, and G. Cybenko. Neural networks, backpropagation and automatic differentiation. In A. Griewank and G. F. Corliss, editors, *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, pages 31–42. SIAM, Philadelphia, PA, 1991.

[22] Z. Tang and G. J. Koehler. Deterministic global optimal FNN training algorithms. *Neural Networks*, 7(2):301–311, 1994.

[23] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[24] P. P. van der Smagt. Minimisation methods for training feedforward neural networks. *Neural Networks*, 7(1):1–11, 1994.

[25] P. J. Werbos. Backpropagation through time:What it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[26] Q. Zhang and A. Benveniste. Wavelet networks. *IEEE Transactions on Neural Networks*, 3(6):889–898, 1992.