

A Learning Result for Recurrent Neural Networks*

Eduardo Sontag
Dept. of Mathematics, Rutgers University
New Brunswick, NJ 08903

sontag@control.rutgers.edu

Abstract

The following learning problem is considered, for continuous-time recurrent neural networks having sigmoidal activation functions. Given a “black box” representing an unknown system, measurements of output derivatives are collected, for a set of randomly generated inputs, and a network is used to approximate the observed behavior. It is shown that the number of inputs needed for reliable generalization (the sample complexity of the learning problem) is upper bounded by an expression that grows polynomially with the dimension of the network and logarithmically with the number of output derivatives being matched.

1 Introduction

This paper is concerned with systems defined by equations of the following type:

$$\dot{x}(t) = \vec{\sigma}^{(n)}(Ax(t) + Bu(t)), \quad y(t) = Cx(t), \quad (1)$$

where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$, and $\vec{\sigma}^{(n)} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is the diagonal map

$$\vec{\sigma}^{(n)} : \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \mapsto \begin{pmatrix} \sigma(x_1) \\ \vdots \\ \sigma(x_n) \end{pmatrix}, \quad (2)$$

and $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ is a Lipschitz continuous mapping (soon to be specialized to $\sigma = \tanh$, the function that most often appears in neural network theory and practice). The dot indicates time-derivative; for simplicity, we later omit the arguments t and the superscript (n) , writing just $\dot{x} = \vec{\sigma}(Ax + Bu)$ and $y = Cx$. We call any system of this general form an *n-dimensional, m-input, p-output recurrent net with activation σ* , or, sometimes, just a “net”. The spaces \mathbb{R}^m , \mathbb{R}^n , and \mathbb{R}^p are called respectively the input-value, state, and output-value spaces. In system-theoretic terms, we may represent a net by means of a block diagram as in Figure 1.

Recurrent nets — sometimes in their discrete-time variant in which one has a difference equation $x(t+1) = \vec{\sigma}(Ax(t) + Bu(t))$ instead of a differential equation — have been proposed as models for control, computation, and signal processing by many researchers, see e.g. [12, 14, 13, 15, 3, 16, 4, 21, 22], as well as in optimization and associative memory design (“Hopfield nets”),

*Supported in part by US Air Force Grant AFOSR-94-0293

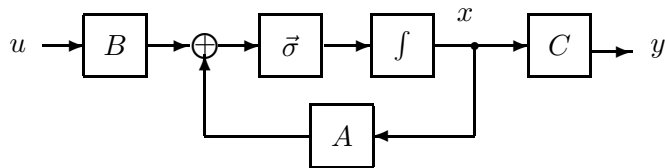


Figure 1: *Block diagram of net*

language inference, and sequence extrapolation for time series prediction. In neural network studies, each coordinate x_i of x is thought of as the internal state of the i th “neuron” in a network. Partial mathematical justification for the interest in the class of systems represented by (1) lies in the fact that they can, in a certain sense, approximate arbitrary nonlinear systems, they provide universal models of digital as well as analog computation, and their system-theoretic properties (controllability, observability, minimality, parameter identifiability, etc) can be quite elegantly characterized (see e.g. [1, 2, 19, 18]).

The Learning Problem

We formulate the problem of identification from input/output data in the paradigm of “probably approximately correct” (PAC) learning, due independently to Valiant in the computer science literature and, in somewhat different but essentially equivalent form, to Vapnik in the statistics literature. An excellent reference is the recent monograph [20].

The question that we address can be described in very intuitive terms as follows. Given is a “black box” or “plant” as in Figure 2. This represents a system (for example, an object to be

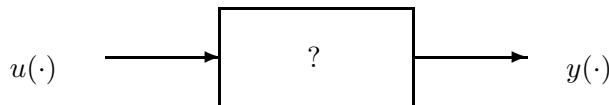


Figure 2: *“Black box” or “plant”*

controlled) whose structure is unknown, but on which it is possible to perform experiments, by applying various inputs u and observing the ensuing outputs y . Assume that we first collect a finite amount of experimental input/output data in this manner, and we subsequently “fit” some net Σ_0 as well as possible to this data, through minimization of a least-squares error criterion $E(\Sigma)$ which penalizes the lack of agreement between the outputs produced by a candidate net Σ and the experimental (“training”) data. The main question here is: how accurate is the obtained Σ_0 as a behavioral model of the plant? In other words, how reliable is Σ_0 as a predictor of the output y that the plant would produce in response to a randomly chosen future input u ? We assume stationarity: both the training samples and the data seen in the future (testing data) are randomly drawn according to the *same* probability distribution.

Since it may be impossible to model the plant behavior exactly by a net, the least-squares minimization of $E(\Sigma)$ will result in some minimal error value $E_0 = E(\Sigma_0)$. If the training data was sufficiently rich, we may then expect the error on future inputs u to be, on the average, close to E_0 (by the law of large numbers applied to the error on each data point, seen as a random variable). One of the main questions in PAC learning is that of quantifying what is a “sufficiently rich” set of training data so that the expected error is indeed “close” (also in a precisely quantified sense) to the training error. Mathematically, the techniques involve uniform

laws of large numbers, providing an elegant generalization of classical questions regarding the uniform convergence of empirical probabilities of sets, and in particular the Glivenko-Cantelli Lemma, and more generally the uniform convergence of empirical means of random variables.

Actually, rather than dealing only with the case where the training data pairs $(u(\cdot), y(\cdot))$ are generated by a (deterministic) plant, one allows as well a more general setup in which it is only postulated that there is a probability distribution P on the possible pairs which may appear. If, as in the previous discussion, there is a plant, let's call it H , which generates the data, and inputs u are generated randomly according to a probability distribution P_0 , there is a natural probability P induced on pairs (u, y) : (u, y) has the same probability as u if $y = H(u)$, and has zero probability otherwise (of course, we are being extremely informal right here, and one must argue in terms of measures). The framework allows, in addition, for stochastic plants as well noisy measurements. These issues are discussed at length in [20].

We will state and prove a precise result on learning in the sense just described. The criterion we use penalizes the differences between the first k derivatives of outputs at time $t = 0$ (corresponding very roughly to “fitting” a model to order $O(T^k)$ on an interval of length T). The estimates of the amount of training data required for reliable prediction grow with the state-space dimension of the nets used as models as well as with the number k of derivatives. (Intuitively, if we use nets of larger dimension, we will be able to fit the training data better, but prediction ability will decrease, as we are essentially memorizing all data, with no “smoothing”. Likewise, if we want to accurately predict more derivatives, we need more data.) For simplicity of exposition, we take the input and output dimensions m and p to be both equal to one, but the statements and proofs can be easily generalized to arbitrary m and p . Pick any infinitely differentiable function σ (soon we will specialize to the case $\sigma = \tanh$).

Let Σ be a net with activation σ and let $\xi \in \mathbb{R}^n$ be any state. For any input function $u : [0, T] \rightarrow \mathbb{R}^m$ and any initial state $\xi \in \mathbb{R}^n$, there is a unique solution $x : [0, T] \rightarrow \mathbb{R}^n$ of $\dot{x} = \vec{\sigma}(Ax + Bu)$ with $x(0) = \xi$; this solution will be denoted as $x(t, \xi, u)$. (By “input function” we mean, as usual, cf. [17], a measurable essentially bounded function, but for the purposes of this paper, differentiable inputs would suffice.) We also consider the ensuing output function $y(t, \xi, u) := Cx(t, \xi, u)$. Observe that the k th derivative of the output at time 0 is a function only of the first $k-1$ derivatives of the input; for instance, $y'(0, \xi, u) = C\vec{\sigma}^{(n)}(A\xi + Bu(0))$. That is, there is for each k a map

$$y_{k, \Sigma, \xi} : \mathbb{R}^k \rightarrow \mathbb{R} \tag{3}$$

such that $y^{(k)}(0, \xi, u) = y_{k, \Sigma, \xi}(u(0), u'(0), \dots, u^{(k-1)}(0))$ for every $(k-1)$ -times differentiable input u . (For $k = 0$, $y_{0, \Sigma, \xi} = C\xi$ is a constant.) We also introduce, for each k , the vector of derivatives of order $\leq k$, writing

$$Y_{k, \Sigma, \xi}(\mu_0, \dots, \mu_{k-1}) := (y_{0, \Sigma, \xi}, y_{1, \Sigma, \xi}(\mu_0), \dots, y_{k, \Sigma, \xi}(\mu_0, \dots, \mu_{k-1})).$$

For each k , we let $\mathcal{Z}_k = \mathbb{R}^k \times \mathbb{R}^{k+1}$ (if $k = 0$, just \mathbb{R}); thus, if $\mu \in \mathbb{R}^k$ then $(\mu, Y_{k, \Sigma, \xi}(\mu)) \in \mathcal{Z}_k$. For each pair $(\mu, \eta) \in \mathcal{Z}_k$, we denote

$$\text{Loss}_{\Sigma, \xi, k}[\mu, \eta] := \frac{\|Y_{k, \Sigma, \xi}(\mu) - \eta\|^2}{1 + \|Y_{k, \Sigma, \xi}(\mu) - \eta\|^2}.$$

This penalizes the mismatch between the data point (μ, η) (which we may interpret as an input/output pair associated to the plant in Figure 2) and the output that the net Σ , initialized at the state ξ , produces in response to the input whose first $k-1$ derivatives are specified by the vector μ . (The error is normalized, for technical reasons, to the range $[0, 1]$.)

Two quantities are of special interest, still for any fixed initialized net (Σ, ξ) . The first is the “empirical risk” or “average training error” obtained by averaging over a finite set of data: for each positive integer s and each $\vec{z} = (z_1, \dots, z_s) \in \mathcal{Z}_k^s$:

$$\text{EmpLoss}_{\Sigma, \xi, k}[\vec{z}] := \frac{1}{s} \sum_{i=1}^s \text{Loss}_{\Sigma, \xi, k}[z_i].$$

The second is defined once that a probability measure P has been fixed on $(\mathcal{Z}_k, \mathcal{B}_k)$, where \mathcal{B}_k is the algebra of Borel sets in \mathcal{Z}_k : it is the P -expectation of the error (viewing $\text{Loss}_{\Sigma, \xi, k}$ as a random variable on \mathcal{Z}_k):

$$\text{ExpLoss}_{\Sigma, \xi, k} := E(\text{Loss}_{\Sigma, \xi, k}).$$

We let \mathcal{S}_n denote the set of all n -dimensional initialized nets with activation $\sigma = \tanh$ (and $m = p = 1$). For any integers $n \geq 5$ and k and any two real numbers $\varepsilon, \delta \in (0, 1)$, we denote the “sample complexity”:

$$S(n, k, \varepsilon, \delta) := \frac{640}{\varepsilon^2} \left(n^6 + n^3 \log_2 k \right) \left(4 + \ln \left(\frac{1}{\varepsilon \delta} \right) \right).$$

For $n < 5$, we define $S(n, k, \varepsilon, \delta)$ using “5” instead of n in the expression above. This is proved later in the paper (P^s means probability with respect to the product measure P^s on \mathcal{Z}_k^s , corresponding to independent identically distributed choices for the input/output pairs z_1, \dots, z_s):

Theorem 1 *For each $n, k, \varepsilon, \delta > 0$ and each $s \geq S(n, k, \varepsilon, \delta)$,*

$$P^s \left(\left| \text{EmpLoss}_{\Sigma, \xi, k}[\vec{z}] - \text{ExpLoss}_{\Sigma, \xi, k} \right| < \varepsilon, \forall (\Sigma, \xi) \in \mathcal{S}_n \right) \geq 1 - \delta.$$

One interpretation of this result is as follows. Assume that we are trying to obtain a simulation model for the unknown plant in Figure 2, assumed to be strictly causal. We use as models n -dimensional nets, and are able to experiment by applying polynomial inputs of degree $< k$, and collecting data on the first k derivatives, at $t=0$, of the output. Let’s write $H(\mu)$ for the vector of output derivatives corresponding to an input whose derivatives are given by μ . A set of $s = S(n, k, \varepsilon, \delta)$ i.i.d. inputs are generated at random, according to some probability distribution P_0 on \mathbb{R}^k , and the derivatives $\eta = H(\mu)$ are obtained for each input in this set of s inputs. There results a vector of observations, or “training set” \vec{z} . Note that the distribution P_0 induces a probability measure P on the set of derivatives of input/output pairs \mathcal{Z}_k in a natural way: it is concentrated on those pairs (μ, η) for which $\eta = H(\mu)$, and $P(\{(\mu, H(\mu)), \mu \in U\}) = P_0(U)$. The training set \vec{z} is distributed according to P . Suppose that we next minimize, over all n -dimensional nets and all possible initial states, the quantity $\text{EmpLoss}_{\Sigma, \xi, k}[\vec{z}]$, obtaining a “best possible fit” (Σ, ξ) , which produces a minimal error E_0 . (Of course, this minimization is itself a hard numerical problem, viz. methods for so-called “recurrent backpropagation” in the neural net literature.) Assume that we next generate a new input u , with derivatives μ , again according to P_0 , and calculate the error $\text{Loss}_{\Sigma, \xi, k}[\mu, H(\mu)]$, which penalizes the mismatch with the prediction that (Σ, ξ) would make. With high confidence (namely, better than $1 - \delta$), we can say that the expected value of $\text{Loss}_{\Sigma, \xi, k}[\mu, \eta]$ is at most $\varepsilon + E_0$. In particular, if we had managed to fit the data perfectly ($E_0 = 0$), then we are assured, with high confidence, that the expected value of the error is at most ε .

2 Proof of Theorem

Fix two positive integers n and k , to be used for dimension of nets and number of output derivatives to be matched, respectively. We let $\sigma = \tanh$ and $m = p = 1$.

2.1 Formulas for Output Derivatives

We will be thinking of the k th derivative of the output at time 0, $y^{(k)}(0, \xi, u)$, as a function of the first $k-1$ derivatives of the input at $t=0$ as well as those parameters (entries of A , B , and C , and coordinates of the initial state ξ) which characterize the particular n -dimensional initialized net (Σ, ξ) being considered. We next formalize this functional dependence. As a first step, we consider the following question: if $x(\cdot)$ satisfies (1), what differential equations do the derivatives of $x(\cdot)$ satisfy? In order to answer this question, we next define certain polynomials \mathcal{X}_j^i , in terms of which the derivatives will be expressed.

We introduce indeterminates $\nu_r, r \geq 1$, $\rho_r, r \in \{1, \dots, n\}$, $\beta_r, r \in \{1, \dots, n\}$, and $\alpha_{rs}, (r, s) \in \{1, \dots, n\}^2$, and define polynomials \mathcal{X}_j^i , for $j \in \{1, \dots, n\}$ and $i \geq 0$, as follows, inductively on i :

$$\mathcal{X}_j^1 := \rho_j$$

and for $i \geq 1$:

$$\mathcal{X}_j^{i+1} := \xi_1 + \dots + \xi_n + \eta_1 + \dots + \eta_{i-1}$$

where we are denoting:

$$\xi_l = \frac{\partial \mathcal{X}_j^i}{\partial \rho_l} \cdot (1 - \rho_l^2) \cdot \left(\sum_{h=1}^n \alpha_{lh} \rho_h + \beta_l \nu_1 \right)$$

and

$$\eta_l = \frac{\partial \mathcal{X}_j^i}{\partial \nu_l} \cdot \nu_{l+1}.$$

If $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times 1}$, $\xi \in \mathbb{R}^n$, and v_0, \dots, v_{i-1} are real numbers, we denote by $\mathcal{X}_j^i(A, B, \xi, v_0, v_1, \dots, v_{i-1})$ the number obtained by substituting into the polynomial \mathcal{X}_j^i :

$$\rho_s \mapsto \sigma(A_s \xi + b_s v_0), \alpha_{rs} \mapsto a_{rs}, \beta_s \mapsto b_s, \forall r, s \in \{1, \dots, n\}, \nu_s \mapsto v_s, \forall r \geq 1,$$

where a_{rs} , b_s , and A_s denote, respectively, the (r, s) th entry of A , the s th entry of the column vector B , and the s th row of the matrix A . The following Lemma provides an interpretation of the polynomials \mathcal{X}_j^i ; it is easily verified by induction, using the chain rule and the fact that $\sigma'(a) = 1 - (\sigma(a))^2$ when $\sigma = \tanh$:

Lemma 2.1 If Σ is an n -dimensional net with $m = 1$ and $\sigma = \tanh$, and x is a solution of $\dot{x}(t) = \vec{\sigma}(Ax(t) + Bu(t))$, for some input u which is $i-1$ times differentiable, then the i th derivative of the coordinate x_j satisfies:

$$x_j^{(i)}(t) = \mathcal{X}_j^i(A, B, x(t), u(t), u'(t), \dots, u^{(i-1)}(t))$$

for all t . □

Now introduce additional indeterminates $\gamma_1, \dots, \gamma_n$ and let

$$\mathcal{Y}^i := \sum_{j=1}^n \gamma_j \mathcal{X}_j^i, \quad i \geq 1.$$

Finally, if $C \in \mathbb{R}^{1 \times n}$ and A, \dots are as earlier, we denote by

$$\mathcal{Y}^i(A, B, C, \xi, v_0, v_1, \dots, v_{i-1}) := \sum_{j=1}^n c_j \mathcal{X}_j^i(A, B, \xi, v_0, v_1, \dots, v_{i-1})$$

the substitution into these variables. Clearly:

Corollary 2.2 If Σ is an n -dimensional net with $m = p = 1$ and $\sigma = \tanh$, and u is an input which is $k-1$ times differentiable, $k \geq 1$, then the k th derivative of the output y at $t = 0$ satisfies:

$$y^{(k)}(0, \xi, u) = \mathcal{Y}^k(A, B, C, \xi, u(0), u'(0), \dots, u^{(k-1)}(0))$$

for each initial state $x(0) = \xi \in \mathbb{R}^n$. □

For example, $\mathcal{Y}^2 = \sum_{j=1}^n \gamma_j (1 - \rho^2) (\sum_{h=1}^n \alpha_{jh} \rho + \beta_j \nu_1)$, so

$$y''(0, \xi, u) = \sum_{j=1}^n c_j \left(1 - (\sigma(A_j \xi + b_j u(0)))^2 \right) \left(\sum_{h=1}^n a_{jh} \sigma(A_h \xi + b_h u(0)) + b_j u'(0) \right).$$

Observe that $\mathcal{Y}^k(A, B, C, \xi, u(0), u'(0), \dots, u^{(k-1)}(0))$ is the same as what we earlier called $y_{k, \Sigma, \xi}(u(0), u'(0), \dots, u^{(k-1)}(0))$, if Σ is the net whose matrices are (A, B, C) .

Since, from the recursive definition, the degree of \mathcal{X}_j^{i+1} equals $(\deg \mathcal{X}_j^i - 1) + 4$, the degree of \mathcal{X}_j^i is $3i - 2$ for $i \geq 1$. Thus:

Lemma 2.3 The degree of \mathcal{Y}^k is $3k - 1$ for each $k \geq 1$. □

Remark 2.4 Note that the degree of \mathcal{Y}_k with respect to the “input” variables ν_l is $k - 1$ for each $k \geq 1$ because, inductively, $\deg_{\nu_1, \dots, \nu_i} \mathcal{X}_j^{i+1} = \deg_{\nu_1, \dots, \nu_i} \mathcal{X}_j^i + 1$. Also, the degree of \mathcal{Y}_k with respect to the “sigmoid” variables ρ_j is $2k - 1$ because, inductively, $\deg_{\rho_1, \dots, \rho_n} \mathcal{X}_j^{i+1} = (\deg_{\rho_1, \dots, \rho_n} \mathcal{X}_j^i - 1) + 3$. We do not use these estimates in what follows, but it is likely that the upper bounds to be given later could be made less conservative when using this additional information. □

2.2 Pseudo-Dimension Estimates

We first recall the concepts of Vapnik-Chervonenkis (VC) dimension and, more generally, pseudo- (or Pollard) dimension. Given a set \mathcal{U}_0 and a class \mathcal{F}_0 of functions $\mathcal{U}_0 \rightarrow \{0, 1\}$ (which we think of as a family of binary classifiers), a *dichotomy* on a subset V of \mathcal{U}_0 is any function $\delta : V \rightarrow \{0, 1\}$, and the subset $V \subseteq \mathcal{U}_0$ is *shattered* by \mathcal{F}_0 if each dichotomy on V is the restriction to V of some $f \in \mathcal{F}_0$. The *Vapnik-Chervonenkis (VC) dimension* $\text{vc}(\mathcal{F}_0)$ is the supremum (possibly infinite) of the set of integers κ for which there is some subset $V \subseteq \mathcal{U}_0$ of cardinality κ which can be shattered by \mathcal{F}_0 . We let H be the Heaviside function: $H(x) = 1$ if

$x > 0$ and 0 if $x \leq 0$. Given now a class of functions \mathcal{F} from \mathcal{U} to \mathbb{R} , we introduce, for each $f \in \mathcal{F}$, the function

$$q_f : \mathcal{U}_0 = \mathcal{U} \times \mathbb{R} \rightarrow \{0, 1\} : (u, y) \mapsto H(f(u) - y)$$

as well as the class \mathcal{F}_0 consisting of all such q_f . The *pseudo-dimension* of \mathcal{F} is defined by: $\text{PD}(\mathcal{F}) := \text{VC}(\mathcal{F}_0)$. (This definition is equivalent to the one in [7].)

We now review a basic pseudodimension estimate from [9], specialized to a form useful for our purposes. Let $\sigma = \tanh$. Assume given a function

$$\varphi : \mathbb{R}^l \times \mathbb{R}^h \rightarrow \mathbb{R}_{\geq 0}$$

which can be expressed as:

$$\varphi(\lambda, z) = P(\sigma(R_1(\lambda, z)), \dots, \sigma(R_n(\lambda, z)), \lambda, z), \quad (4)$$

where $R_1(\lambda, z), \dots, R_n(\lambda, z)$ are polynomials each of (total) degree at most $r - 1$ and P is a polynomial of degree at most $q - 1$. We view vectors $\lambda \in \mathbb{R}^l$ as parameters, and for each λ let

$$\varphi_\lambda : \mathbb{R}^h \rightarrow \mathbb{R}_{\geq 0} : z \mapsto \varphi(\lambda, z).$$

Lemma 2.5 Let

$$\mathcal{F} := \left\{ \frac{\varphi_\lambda}{1 + \varphi_\lambda}, \lambda \in \mathbb{R}^l \right\}.$$

Then,

$$\text{PD}(\mathcal{F}) \leq b + 2(1 + \log_2 e)l, \quad (5)$$

where

$$b \leq (ln)^2 - ln + 2[l \log_2 q + l \log_2 l + l \log_2(q + r) + ln \log_2(l^2(q + r))]. \quad (6)$$

Proof. By definition, $\text{PD}(\mathcal{F})$ equals the VC dimension of the class of binary functions $(z, z_0) \mapsto H(\varphi_\lambda(z)/(1 + \varphi_\lambda(z)) - z_0)$, $\lambda \in \mathbb{R}^l$. Since $1 + \varphi_\lambda(z)$ is always positive, we can write this latter expression also as $H(\psi(z, z_0))$, where $\psi(z, z_0) := \varphi_\lambda(z) - z_0(1 + \varphi_\lambda(z))$. Observe that ψ is a polynomial of total degree $\leq (q - 1) + 1 = q$ on λ, z, z_0 and the expressions $\sigma(R_i(\lambda, z))$. On the other hand,

$$\frac{\partial \sigma(R_i(\lambda, z))}{\partial \lambda_j} = (1 - (\sigma(R_i(\lambda, z)))^2) \frac{\partial R_i(\lambda, z)}{\partial \lambda_j}$$

for each coordinate λ_j of λ , and a similar expression holds for partial derivatives with respect to the coordinates of z . Thus, each $\sigma(R_i(\lambda, z))$ is a Pfaffian function of degree $\leq 2 + (r - 2) = r$. The result then follows from [20], Theorem 10.7 and Lemma 10.6, which provide an upper bound for the VC dimension of binary function classes defined by polynomial expressions involving Pfaffian functions, in terms of the degree of the polynomials and the degree and length of the Pfaffian chains involved. Precisely, we apply Theorem 10.7 with “ b ” instead of “ $2 \log_2 B$ ” and Lemma 10.6 with q, d, D respectively equal to our n, q, r , to obtain respectively equations (5) and (6). (The results cited from the book [20] are based on the paper [9]; the only minor difference is in the slight improvement in (5), which the original paper had given as $b + 16l$.) ■

Some algebraic manipulation gives:

Corollary 2.6 When $r = 3$, $l \leq 3n + n^2$, $n \geq 5$, and $q \leq 6k$, $\text{PD}(\mathcal{F}) \leq 3n^6 + 5n^3 \log_2 k$. □

2.3 Uniform Approximation of Expected Loss

The last ingredient is a theorem on approximation of means by empirical means, which we quote from [7] (see also [20] for an exposition).

Let $(\mathcal{Z}, \mathcal{A})$ be a set together with a σ -algebra of subsets of \mathcal{Z} , and let $\varphi : \mathbb{R}^l \times \mathcal{Z} \rightarrow [0, 1]$, where l is some positive integer, be a function measurable with respect to $\mathcal{B} \times \mathcal{A}$, where \mathcal{B} is the Borel σ -algebra on \mathbb{R}^l . Write $\mathcal{F} := \{\varphi(\lambda, \cdot), \lambda \in \mathbb{R}^l\}$. Assume that $d = \text{PD}(\mathcal{F}) < \infty$.

Then, applying Corollary 2 in Section 4 of [7]:

Proposition 2.7 Let \bar{z} be generated by s independent draws according to any distribution P on $(\mathcal{Z}, \mathcal{A})$. Pick any $\varepsilon, \delta \in (0, 1)$. Then, provided

$$s \geq \frac{64}{\varepsilon^2} \left(2d \ln \left(\frac{16e}{\varepsilon} \right) + \ln \left(\frac{8}{\delta} \right) \right),$$

$$P \left(\forall \varphi \in \mathcal{F} : \left| \widehat{E}_{\bar{z}}(\varphi) - E(\varphi) \right| < \varepsilon \right) \geq 1 - \delta,$$

where

$$\widehat{E}_{\bar{z}}(\varphi) := \frac{1}{s} \sum_{i=1}^s \varphi(z_i)$$

and $E(\varphi)$ denotes the expectation of $\varphi \in \mathcal{F}$. □

2.4 Proof of Theorem 1

Theorem 1 is an immediate consequence of Proposition 2.7 and Corollary 2.6, as we discuss now.

The set of all nets of dimension n and activation tanh, together with initial states, may be parameterized by $\lambda \in \mathbb{R}^l$, where $l := 3n + n^2$, after listing in some specific order the entries of A , B , C , and ξ . Hence we can write, if u is an input which is $i-1$ times differentiable, and $i \geq 1$, $y^{(i)}(0, \xi, u) = y_{i, \Sigma, \xi}(u(0), u'(0), \dots, u^{(i-1)}(0))$ as $\varphi_i(\lambda, u(0), u'(0), \dots, u^{(i-1)}(0))$, for some function $\varphi_i : \mathbb{R}^l \times \mathbb{R}^i \rightarrow \mathbb{R}$. We introduce $\varphi : \mathbb{R}^l \times \mathcal{Z}_k \rightarrow \mathbb{R}_{\geq 0}$ defined by:

$$\varphi(\lambda, ((\mu_0, \dots, \mu_{k-1}), (\eta_0, \dots, \eta_k))) = \sum_{i=0}^k (\varphi_i(\lambda, \mu_0, \dots, \mu_{i-1}) - \eta_i)^2$$

and note that this equals $\|Y_{k, \Sigma, \xi}(\mu) - \eta\|^2$ if (Σ, ξ) is the initialized system corresponding to the parameter λ . Let, for each $1 \in \{1, \dots, n\}$, $R_i(\lambda, z) := \sum_{h=1}^n (\alpha_{ih} \xi_h + \beta_i \mu_0)$. Thus we have the situation in Equation (4), $\varphi(\lambda, z) = P(\sigma(R_1(\lambda, z)), \dots, \sigma(R_n(\lambda, z)), \lambda, z)$, where P has degree $6k - 2$ (by Lemma 2.3), and each R_i has degree 2. So, in the notations there, we may take $r = 3$, $q = 6k - 1$. Corollary 2.6 then gives the estimate $d \leq 3n^6 + 5n^3 \log_2 k$ to be used in Proposition 2.7, and the bound for s can then be estimated to give the simpler expression in Theorem 1.

3 Some Remarks

It should be possible to improve considerably on the learning result, which should be only seen as a first step in the development of a serious study of identification complexity using the

computational learning theory paradigm. It is too conservative not only in the numeric constant (see other estimates in [20]) but also in the fact that it is a “worst possible case” result, valid with respect to all possible probability distributions, and in its use of pseudodimension estimates, which are in general conservative.

It is worth remarking that it is also possible to give a simple sample complexity estimate for linear systems identification, i.e., the case when σ is the identity, as well as some other nonlinear situations (e.g., if σ is a piecewise-polynomial function). All these follow from pseudodimension estimates. For example, for the class of minimal linear systems of dimension n , we obtain

$$\text{PD}(\mathcal{F}) = O(n \log k).$$

This follows from the estimates for VC dimension of algebraic concept classes given in [6] (see [20], Theorem 10.5 or Corollary 10.2), since ψ in the proof of Lemma 2.5 has degree $2k + 5$ and one may take $l = 2n$ (because, by linear systems theory, reachable and observable systems can be parameterized linearly using that many parameters).

We have avoided discussion of discrete-time networks. For discrete-time systems there exist *lower bounds* on pseudodimension; see [5] for the case of linear systems and [10] for discrete time nonlinear nets. Interestingly enough, for $\sigma = \tanh$, the lower bounds are at least linear (rather than logarithmic) on k , for discrete-time systems. This surprising difference is related to the fact that, in discrete-time, outputs at time k require k compositions of sigmoids, while in continuous time, using derivatives of outputs as data, we were able to avoid compositions.

References

- [1] Albertini, F., and E.D. Sontag, "For neural networks, function determines form," *Neural Networks* **6**(1993): 975-990.
- [2] Albertini, F., and E.D. Sontag, "State observability in recurrent neural networks," *Systems & Control Letters* **22**(1994): 235-244.
- [3] Back, A.D., and A.C. Tsoi, "FIR and IIR synapses, a new neural network architecture for time-series modeling," *Neural Computation* **3** (1991): 375-385.
- [4] Bengio, Y., *Neural Networks for Speech and Sequence Recognition*, Thompson Computer Press, Boston, 1996.
- [5] Dasgupta, B., and E.D. Sontag, "Sample complexity for learning recurrent perceptron mappings," *IEEE Trans. Inform. Theory* **42** (1996): 1479-1487.
- [6] Goldberg, P., and M. Jerrum, "Bounding the Vapnik-Chervonenkis dimension of concept classes parametrized by real numbers," *Machine Learning* **18**(1995): 131-148.
- [7] Haussler, D., "Decision theoretic generalizations of the PAC model for neural nets and other learning applications", *Information and Computation* **100**(1992): 78-150.
- [8] Hautus, M., "A set of IP-functions," unpublished manuscript, Eindhoven University, August 1993.
- [9] Karpinski, M., and A. Macintyre, "Polynomial bounds for VC dimension of sigmoidal and general Pfaffian neural networks," *J. Computer Sys. Sci.*, to appear. (Summary in "Polynomial bounds for VC dimension of sigmoidal neural networks," in *Proc. 27th ACM Symposium on Theory of Computing, 1995*, pp. 200-208.)
- [10] Koiran, P., and E.D. Sontag, "Vapnik-Chervonenkis dimension of recurrent neural networks," *Proceedings of Third European Conference on Computational Learning Theory*, Jerusalem, March 1997, to appear.
- [11] Leshno, M., V.Ya. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a non-polynomial activation function can approximate any function," *Neural Networks* **6**(1993): 861-867.
- [12] Matthews, M., "A state-space approach to adaptive nonlinear filtering using recurrent neural networks," *Proc. 1990 IASTED Symp. on Artificial Intelligence Applications and Neural Networks*, Zürich, July 1990, pp. 197-200.
- [13] Polycarpou, M.M., and P.A. Ioannou, "Neural networks and on-line approximators for adaptive control," in *Proc. Seventh Yale Workshop on Adaptive and Learning Systems*, pp. 93-798, Yale University, 1992.
- [14] Siegelmann, H.T., and E.D. Sontag, "Turing computability with neural nets," *Appl. Math. Lett.* **4**(6)(1991): 77-80.
- [15] Sontag, E.D., "Neural nets as systems models and controllers," in *Proc. Seventh Yale Workshop on Adaptive and Learning Systems*, pp. 73-79, Yale University, 1992.

- [16] Sontag, E.D., “Neural networks for control,” in *Essays on Control: Perspectives in the Theory and its Applications* (H.L. Trentelman and J.C. Willems, eds.), Birkhauser, Boston, 1993, pp. 339-380.
- [17] Sontag, E.D., *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, Springer, New York, 1990.
- [18] Sontag, E.D., “Recurrent neural networks: some systems-theoretic aspects,” to appear.
- [19] Sontag, E.D., and H.J. Sussmann, “Complete controllability of continuous-time recurrent neural networks,” *Systems and Control Letters*, 1997, to appear.
- [20] Vidyasagar, M., *A Theory of Learning and Generalization: With Applications to Neural Networks and Control Systems*, Springer, London, 1997.
- [21] Zbikowski, R., “Lie algebra of recurrent neural networks and identifiability,” *Proc. Amer. Automatic Control Conference*, San Francisco, 1993, pp. 2900-2901.
- [22] Zbikowski, R., and K.J. Hunt, eds., *Neural Adaptive Control Technology* World Scientific Publishing, 1996.