Appl. Math. and Comp. Sci., 1994, vol.4, No.3, 337–342

LEARNING THROUGH INTEGRAL REPRESENTATIONS

MAREK B. ZAREMBA*, EUGENIUSZ PORADA*

This paper addresses the issue of building neural networks capable of approximating arbitrary continuous target functions. An approach based on integral representation of the desired function is presented. The method allows the user to construct the network architecture by applying qualitative geometrical analysis of the space of input states.

1. Introduction

Function approximation capabilities of connectionist networks have been studied by several authors (Funahasi, 1989; Hornik *et al.*, 1989; Irie and Miyake, 1988). It has been proved *inter alia* that the networks can approximate an arbitrary continuous target function uniformly on compact domains in Euclidean space. The mathematical analysis has essentially been based on the Kolmogorov theorem known as the negative solution of the 13th problem of Hilbert. The theoretical complexity involved does not allow us to design a constructive method of learning from the mathematical demonstrations. A more promising method is to find the exact integral representation of a desired function (with the use of a continuum of processing units (Irie and Miyake, 1988)) and then approximate the integral by finite subsets of the continuum. The condition of constructiveness requires specific integral representations of the target function; an explicit integral has to be found that will allow for uniform convergence of its discrete approximations. This paper is a first step in the development of such explicit regular integrals representing a target function.

The learning method has been oriented especially for use in measurement systems (Bock *et al.*, 1992; Zaremba *et al.*, 1991). In such systems, a neural processor extracts a measurand (the current value of a physical parameter) from a sensor distributed signal. The input to the processor can be looked upon as a vector running through a one-parameter manifold in a Euclidean space. Thus, the problem arises of uniformly approximating, with a given precision, a target function defined on a one-dimensional manifold.

2. Connectionist Architecture

Vector $\boldsymbol{x} \in \mathbb{R}^N$ constitutes the input to the connectionist network under consideration. The first layer of connections between N input units and n hidden units converts the input signals into hidden signals that are fed to the layer of hidden units. Thus, the input to a hidden unit has the form

$$H = \phi(\boldsymbol{x} \cdot \boldsymbol{m} - \beta)$$

* Département d'informatique, Université du Québec à Hull, 101 St-Jean Bosco, Hull, Québec, J8Y 3G5, Canada

where ϕ is the transfer function of the hidden unit, β denotes its bias coefficient, and $\boldsymbol{m}$ is the N–vector of connection weights between the input units and the hidden unit. We assume a transfer function of the form

$$\phi(x) = \begin{cases} x & \text{if} \quad x > 0 \\ 0 & \text{if} \quad x \leq 0 \end{cases}$$

The bias of a hidden unit can be expressed as

$$\beta = \boldsymbol{b} \cdot \boldsymbol{m}$$

where $\boldsymbol{b}$ is a fixed N–vector. Consequently, the hidden signal can be written as the following function of $\boldsymbol{x}$:

$$H_{m,b}(\boldsymbol{x}) = \begin{cases} (\boldsymbol{x} - \boldsymbol{b}) \cdot \boldsymbol{m} & \text{if} \quad (\boldsymbol{x} - \boldsymbol{b}) \cdot \boldsymbol{m} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The hidden unit itself will be denoted by $(\boldsymbol{m}, \boldsymbol{b})$.

The second layer of connections combines the hidden signals into a numerical output signal, generated by the output unit. Thus, the network performs the following mapping, called the *output function*:

$$F_{\text{out}}(\boldsymbol{x}) = \sum_{j=1}^{n} H_j(\boldsymbol{x}) \mu_j$$

where n is the number of hidden units and μ_j denotes the weight of the connection between hidden unit j and the output unit. The problem of function approximation by connectionist networks consists in constructing a hidden layer and defining weights μ_j such that $F_{\text{out}}(\boldsymbol{x})$ approximates a target function $f(\boldsymbol{x})$ uniformly in the domain X with a desired precision.

3. Integral Representation of the Target Function

A method for construction of connectionist networks approximating a target function and consisting in the assumption of a continuum of hidden units was proposed in (Irie and Miyake, 1988). Assuming a continuum of hidden units, we consider the *integral output function*:

$$I_{\text{out}}(\boldsymbol{x}) = \int_{\mathbb{K}} H_k(\boldsymbol{x}) \mathrm{d}\mu(k)$$

where $\mathbb{K}$ is the continuum, $H_k(\boldsymbol{x})$ represents the hidden signal produced by hidden unit k, and μ denotes a weight distribution in $\mathbb{K}$.

It is proved in (Hornik *et al.*, 1989) that any sufficiently regular target function (a continuous function $f(\boldsymbol{x})$ defined on a compact set $X \subset \mathbb{R}^N$, and thus a uniformly continuous target function), can be exactly represented by a continuum of hidden units:

$$f(\boldsymbol{x}) = I_{\text{out}}(\boldsymbol{x}), \qquad \boldsymbol{x} \in X$$

The integral representation is defined indirectly by means of Fourier transform. Our method aims at directly constructed finite measures and explicit integrals. In this paper we report results for the case where X is a one-dimensional compact manifold in $\mathbb{R}^N$.

Let us consider the implications of the representation theorems when it comes to the function approximation capabilities of the connectionist networks. A crucial result would be an explicit form of the measure μ. This can be used in constructive procedures for finding discrete approximations of the exact integral representation: we can imitate the approximations of a Riemann integral by the discrete Riemann sums. In this way, for a given $\boldsymbol{x} \in X$, we get finite linear combinations of appropriately selected hidden signals which represent the output function at point $\boldsymbol{x}$ as closely as required. This statement still holds true for any finite family of points $\boldsymbol{x}$. By the uniform continuity of the integral output function, $F_{\text{out}}(\boldsymbol{x})$ approximates the integrals $I_{\text{out}}(\boldsymbol{x})$ uniformly in any given compact domain $X \subset \mathbb{R}^N$. So, finally, the connectionist networks approximate arbitrary continuous functions defined in compact domains.

4. Geometrical Method for Integral Representations

In this section, we outline the integral representation theorem for the case where X is a regular arc in $\mathbb{R}^N$, i.e. a parametric differentiable curve without loops:

$$\boldsymbol{x} = \boldsymbol{x}(s) \in \mathbb{R}^N, \quad \boldsymbol{x}(s_a) \neq \boldsymbol{x}(s_b) \quad \text{for} \quad s_a \neq s_b, \quad \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s} \neq 0 \quad \text{for} \quad s \in [s_0, s_1]$$

Consider an arbitrary target function f defined on X. We express the target function by means of the variable s and assume that $f(s)$ is a smooth function.

Now we construct an integral representation of the target function, assuming that the parametric curve $\boldsymbol{x}(s)$ satisfies the so-called convex separability condition. The definition of the separability involves partitioning of the curve into the two following arcs, namely the arc

$$A_\sigma = \{\boldsymbol{x}(s) : s \leq \sigma\}$$

and the complementary arc

$$B_\sigma = X \setminus A_\sigma = \{\boldsymbol{x}(s) : s > \sigma\}$$

The curve meets the convex separability condition if

$$\text{conv}(A_\sigma) \cap B_\sigma = \emptyset, \qquad s_0 \leq s \leq s_1$$

where $\text{conv}(A)$ denotes the convex hull of a set $A \subset \mathbb{R}^N$. Notice that a planar spiral curve or a helix in $\mathbb{R}^3$ will satisfy the condition. In fact, any geometrical smooth curve in $\mathbb{R}^N$ satisfies the convex separability condition if we allow for discontinuous parametric representations that are piecewise regular. In the case of piecewise regularity, the method described below is applied for each regular arc of the geometrical curve. The general method involves extensive geometrical procedures which we will not go into in detail in this paper. However, for the planar curves, we developed

numerical methods for creating splines to combine local approximation functions into a global approximation function.

For a given $\sigma \in [s_0, s_1]$, let $\boldsymbol{g}_\sigma$ be a vector othogonal to the hyperplane tangent to $\mathrm{conv}(A_\sigma)$ at point $\boldsymbol{x}(\sigma)$; the length and the sense of $\boldsymbol{g}_\sigma$ is determined by the condition

$$\left.\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s}\right|_\sigma \cdot \boldsymbol{g}_\sigma = 1 \tag{2}$$

The tangent hyperplanes are not uniquely defined, so the vector function

$$\sigma \to \boldsymbol{g}_\sigma$$

called the *generic function*, can be constructed in different manners. For planar curves fulfilling the convex separability condition, we developed numerical methods for finding continuous generic functions (such that the initial value $\boldsymbol{g}_{s_0}$ is co-linear with the derivative $\mathrm{d}\boldsymbol{x}/\mathrm{d}s$ at $s = s_0$).

Let σ denote the hidden unit $(\boldsymbol{g}_\sigma, \boldsymbol{x}(s))$. We use the σ units for the integral representation of our target function $f(s)$, so $\mathbb{K} = [s_0, s_1]$. In this way, the generic function is expressed as a continuous function defined on a compact topological space.

Now we determine the weight distribution $\mu(\sigma)$. Mathematically, μ is a measure (not necessarily positive) containing a continuous component μ_c (a measure vanishing at points) and a discrete component μ_d (a measure accumulated in a discrete set). Thus, the continuous and the discrete components of the representation will be constructed.

First notice that $(\boldsymbol{x}(s) - \boldsymbol{x}(\sigma)) \cdot \boldsymbol{g}_\sigma \leq 0$ for $\sigma \geq s$ and that there exists a positive δ such that $(\boldsymbol{x}(s) - \boldsymbol{x}(\sigma)) \cdot \boldsymbol{g}_\sigma > 0$ when $s - \delta < \sigma < s$. Using compactness arguments one can prove that a positive δ independent from the point s can be selected. Thus, according to (1), we have for all s

$$\begin{aligned} &\text{i)} \quad H_a(\boldsymbol{x}(s)) = (\boldsymbol{x}(s) - \boldsymbol{x}(\sigma)) \cdot \boldsymbol{g}_\sigma \quad \text{if} \quad s - \delta < \sigma < s \\ &\text{ii)} \quad H_a(\boldsymbol{x}(s)) = 0 \qquad\qquad\qquad\quad \text{if} \quad \sigma \geq s \end{aligned} \tag{3}$$

The construction of weight distribution on $\mathbb{K}$ is accomplished by locally extending both the distribution μ and the domain where the output function exactly represents the target function. Thus, we extend $\mu(s)$ and the exact representation on segment $[a, a+\delta]$, assuming that weights $\mu(\sigma)$ are already defined in $[s_0, a]$ and $f(s) = F(s)$ for $s \leq a$, where $F(s)$ is the current output function. Under condition ii) in (3), hidden units $\sigma > a$ do not modify the output function at points $s < a$, so, for the purpose of the extension, it is sufficient to find $\mu(\sigma)$, $a \leq \sigma < a + \delta$ such that:

$$\mu(a) H_a(\boldsymbol{x}(s)) + \int_a^{a+\delta} H_a(\boldsymbol{x}(s))\, \mathrm{d}\mu(\sigma) = f(s) - F(s), \quad a \leq s < a + \delta$$

We create the discrete component by setting the weight $\mu(a)$ to the value of $\left.\frac{\mathrm{d}(f-F)}{\mathrm{d}s}\right|_{a+}$. The continuous component will represent the function

$$f^*(s) = f(s) - F(s) - \mu_a H_a(\boldsymbol{x}(s)), \qquad f^*(a) = \left.\frac{\mathrm{d}f^*}{\mathrm{d}s}\right|_a = 0$$

Now, the problem is to find a continuous distribution $\mu(\sigma)$, $a < \sigma < a+\delta$, such that

$$\int_a^{a+\delta} H_a(\boldsymbol{x}(s))\,\mathrm{d}\mu(\sigma) = f^*(s), \qquad a < s < a+\delta \tag{4}$$

The following theorem gives a solution. In what follows we will write interchangeably $\frac{\mathrm{d}}{\mathrm{d}s}\big|_\sigma$ or $\frac{\mathrm{d}}{\mathrm{d}\sigma}$. The outer (tensor) product of vectors will be denoted by $\times$. We will use the following general formula: $[\boldsymbol{a}\times\boldsymbol{b}]\boldsymbol{c} = (\boldsymbol{b}\cdot\boldsymbol{c})\boldsymbol{a}$.

Theorem 1. *Consider the solution $\boldsymbol{v}(\sigma)$ of the linear differential equation*

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}\sigma} + \left[\boldsymbol{g}_\sigma \times \frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}\sigma^2}\right]\boldsymbol{v}(\sigma) = \frac{\mathrm{d}^2 f^*}{\mathrm{d}\sigma^2}\boldsymbol{g}_\sigma$$

with the initial condition $\boldsymbol{v}(a) = 0$. The derivative $\mathrm{d}\boldsymbol{v}/\mathrm{d}\sigma$ is co-linear with $\boldsymbol{g}_\sigma$:

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}\sigma} = \mu(\sigma)\boldsymbol{g}_\sigma$$

where the function $\mu(\sigma)$ fulfils (4).

This theorem shows that the continuous component of an integral representation can be generated by means of linear differential equations. This is particularly important in neural network practice: various well established methods of approximative solutions (such as the Picard method of successive approximations or the method of Euler tangents) can be applied as learning methods. Moreover, the regularity of exact solutions ensures good convergence of discrete representations to the integral representation, allowing for construction of optimal connectionist networks approximating a given target function with a desired precision.

Proof of Theorem 1. Since $\left[\boldsymbol{g}_\sigma \times \frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}\sigma^2}\right]\boldsymbol{v}(\sigma) = \left(\frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}\sigma^2}\cdot\boldsymbol{v}(\sigma)\right)\boldsymbol{g}_\sigma$ we have

$$\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}\sigma} = \mu(\sigma)\boldsymbol{g}_\sigma, \qquad \text{where} \quad \mu(\sigma) = \frac{\mathrm{d}^2 f^*}{\mathrm{d}\sigma^2} - \frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}\sigma^2}\cdot\boldsymbol{v}(\sigma)$$

Thus

$$\frac{\mathrm{d}^2 f^*}{\mathrm{d}\sigma^2} = \mu(\sigma) + \frac{\mathrm{d}^2\boldsymbol{x}}{\mathrm{d}\sigma^2}\cdot\boldsymbol{v}(\sigma) = \frac{\mathrm{d}}{\mathrm{d}\sigma}\left(\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\sigma}\cdot\boldsymbol{v}(\sigma)\right)$$

The second equality can be checked by direct computation, using (2). Thus

$$\left.\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s}\right|_\sigma \cdot\,\boldsymbol{v}(\sigma) = \frac{\mathrm{d}f^*}{\mathrm{d}\sigma}$$

since the two members coincide at $\sigma = a$ and have equal derivatives. The left-hand member, in turn, is a derivative of the expression

$$I(s) = \int_a^s [\boldsymbol{x}(s) - \boldsymbol{x}(\tau)]\cdot\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}\tau}\,\mathrm{d}\tau$$

In fact,

$$\begin{aligned}\frac{\mathrm{d}I}{\mathrm{d}s} &= \frac{\mathrm{d}}{\mathrm{d}s}\left(\boldsymbol{x}(s)\cdot\int_a^s \mathrm{d}\boldsymbol{v}(\tau) - \int_a^s \boldsymbol{x}(\tau)\cdot\mathrm{d}\boldsymbol{v}(\tau)\right)\\ &= \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s}\cdot\int_a^s \mathrm{d}\boldsymbol{v}(\tau) + \boldsymbol{x}(s)\cdot\left.\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}\tau}\right|_s - \boldsymbol{x}(s)\cdot\left.\frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}\tau}\right|_s\\ &= \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s}\cdot[\boldsymbol{v}(s)-\boldsymbol{v}(a)] = \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}s}\cdot\boldsymbol{v}(s)\end{aligned}$$

having taken into consideration the initial condition. Thus $I(s) = f^*(s)$, because $I(a) = f^*(a) = 0$. On the other hand,

$$I(s) = \int_a^s [\boldsymbol{x}(s)-\boldsymbol{x}(\sigma)]\cdot\boldsymbol{g}_\sigma\mu(\sigma)\,\mathrm{d}\sigma = \int_a^s H_a(\boldsymbol{x}(s))\,\mathrm{d}\mu(\sigma) = \int_a^{a+\delta} H_a(\boldsymbol{x}(s))\,\mathrm{d}\sigma$$

because of (3). The theorem is proved. ■

5. Conclusions

The method of integral representation proposed in this paper makes it possible to construct a neural network for a particular function approximation task. The construction process works efficiently even if the number of hidden processors is fairly limited. This capability is of particular importance in a number of applications, a typical application being in measurement systems. The constructive learning method based on geometrical analysis of the input space and subsequent integral representations has already been proved useful in opto-electronic measurement systems (Bock *et al.*, 1992).

References

Bock W.J., Porada E. and Zaremba M.B. (1992): *Neural processing-type fiber-optic strain sensor.* — IEEE Trans. Instrum. and Measurement, v.41, No.6, pp.1062–1066.

Funahashi K.I. (1989): *On the approximate realization of continuous mappings by neural networks.* — Neural Networks, v.2, No.1, pp.183–192.

Hornik K., Stinchcombe M. and White H. (1989): *Multilayer feedforward networks are universal approximators.* — Neural Networks, v.2, No.1, pp.359–366.

Irie B. and Miyake S. (1988): *Capabilities of three-layered perceptrons.* — IEEE 2nd Int. Conf. *Neural Networks*, San Diego, pp.I:641–648.

Zaremba M.B., Bock W.J. and Porada E. (1991): *The recognition and measurement of optically detected physical variables using interactional neural networks.* — Proc. Int. AMSE Conf. *Neural Networks*, San Diego, USA, v.2, pp.77–85.