NEURAL COMPUTING AND STOCHASTIC OPTIMIZATION

Eugene Wong

Office of Science and Technology Policy, the White House, and
University of California at Berkeley

1 Introduction

One of the promises of neural networks has always been their potential to have the kind of learning and general computing capabilities possessed by the central nervous system. To date, the potential of neural networks to learn has been extensively explored and the results are somewhat encouraging. There exist algorithms that can be used to systematically adjust the parameter of a neural network on the basis of sample inputs for which the desired response of the network is known. Under quite reasonable conditions, the parameters will converge to values such that the network will respond correctly not only to the known samples, but also for unknown inputs that are "similar" to the samples. Whether or not this is "true learning" is immaterial. These characteristics of neural networks render them potentially very useful systems for some classes of problems that do not lend themselves to efficient analytical or algorithmic solution. Viewing neural networks as computers, we might term the learning feature as "programming-by-example."

However, to justify applying the term "computer" to neural networks, some degree of generality is required. Here, the results are less encouraging. To date, the principal applications have been to pattern classification problems of one kind or another. For such problems, the lack of any underlying analytical models makes programming-by-example a particularly attractive way of designing the classifier. But examples of application not of this type are hard to come by.

The purpose of this paper is to examine a class of dynamical neural networks that can be used to find the global optimum for rather general functions. If this approach proves to be practical, then a rather large class of problems, from differential equations to digital communications, can be solved using such networks.

2 Stochastic Optimization

Let $E(x)$, $x \in S$, be a function (called the energy function) to be minimized over some space S. For our purposes, we shall take S to be either the unit cube $[0,1]^n$ or $\{-1,1\}^n$. Now, suppose that we can construct a stationary Markov process X_t, with discrete or continuous time, such that the equilibrium distribution of X_t is given by a density function

$$p(x) = \frac{1}{Z} e^{-\frac{1}{T}E(x)}, \quad x \in S \tag{2.1}$$

where

$$Z = \int_S e^{-\frac{1}{T}E(x)}\,dx \tag{2.2}$$

or

$$Z = \sum_S e^{-\frac{1}{T}E(x)} \tag{2.3}$$

according as whether X_t is continuous or discrete valued. The positive parameter T has the interpretation of temperature. For low temperature, $p(x)$ is sharply peaked at the local minima of $E(x)$. Intuitively, if we take a sequence of temperatures $\{T_k\}$ that decreases slowly enough so that X_t remains in equilibrium distribution, then as $T_k \to 0$, $X(t)$ should end up in the state corresponding to the global minimum of $E(x)$. Indeed, this situation can be made rigorous and for a suitable "cooling schedule" $\{T_k\}$ it can be proved that $X(t)$ will converge to the global minimum of $E(x)$, assuming that the global minimum is unique. The global minimization algorithm that results from this approach is known as *simulated annealing* [KIR83].

3 Dynamical Neural Nets

The states $x(t)$ of a Hopfield net [HOP82] are defined by a pair of equations

$$x_i(t) = g\Big[u_i(t)\Big] \tag{3.1}$$

$$\frac{d\,u_i(t)}{dt} = -E_i\Big[x(t)\Big] \tag{3.2}$$

where $E_i(x) = \frac{\partial}{\partial x_i}E(x)$. If g is a smooth increasing function, then in equilibrium $x(t)$ will be a local minimum of $E(x)$. This is because

$$\frac{d}{dt}E\Big[x(t)\Big] = -\sum_i E_i^2(x(t))\,g'\Big[u_i(t)\Big] \tag{3.3}$$

so that $E(x(t))$ is never increased. In [WON91a] we show that (3.1) can be modified so that $x(t)$ is a stationary Markov process with an equilibrium density function given by (2.1). The modified version of (3.2) is given by a stochastic differential equation

$$d\,u_i(t) = -E_i\Big[x(t)\Big]\,dt + \sqrt{\frac{2T}{g'(u_i(t))}}\,dW_i(t) \tag{3.4}$$

where $\{W_i(t)\}$ is a collection of independent Wiener processes. A popular choice for $g(\cdot)$ is

$$g(u) = \tfrac{1}{2}(1 + \tanh\tfrac{u}{a}) \tag{3.5}$$

for which (3.4) reads

$$d\,u_i(t) = -E_i\Big[x(t)\Big]\,dt + 2\sqrt{aT}\,\cosh\frac{u_i(t)}{a}\,dW_i(t) \tag{3.6}$$

Equation (3.4) can be implemented by injecting which Gaussian into a Hopfield net. We

have called such a realization a *diffusion machine*.

4 Boltzmann Machines

For the discrete version it is convenient to take $E(x)$ to range over $x \in \{-1,1\}^n$ and define

$$\Delta_i(x) = E(\bar{x}_i) - E(x) \tag{4.1}$$

where $\bar{x}_i$ denotes x with x_i replaced by $-x_i$. We have shown [WON91b] that a Markov chain $X(t)$ with an equilibrium distribution given by the Gibbs distribution

$$\text{Prob}(X(t)=x) = \frac{1}{Z} e^{-\frac{1}{T}E(x)} \tag{4.2}$$

can be constructed by injecting noise in a network and identifying $X_i(t)$ with the state at the ith node of the network. For convenience let t take integer values and assume that for each time change $t \to t+1$ only one node can change state, but that as $t \to \infty$ every node changes state infinitely often. Under these assumptions the construction can be described by a transition equation of the form

$$X_i(t+1) = X_i(t)\,\text{sgn}[\Delta_i(X(t)) - Z(t)] \quad \text{for some } i \tag{4.3}$$

$Z(t)$ is a continuous valued stochastic process with independent values at different t's (white noise). It turns out that Z_t cannot be Gaussian, but its density function must be of the form

$$p_Z(z) = -\frac{d}{dz}\left[e^{-\frac{z}{2T}} f(|z|)\right] \tag{4.4}$$

when f is any function that makes p_Z a probability density function. For example, taking

$$f(z) = e^{-\frac{z}{2T}}$$

yields

$$p_Z(z) = \frac{1}{T} e^{-\frac{z}{T}} 1(z)$$

where $1(\cdot)$ denotes the unit-step. We have called a neural network with states governed by (4.3) and (4.4) a *Boltzmann machine*.

5 Digital Receiver — An Example

A potentially important practical application of Boltzmann machines is as an optimum digital receiver. A somewhat simplified model of digital communication can be described as follows:

encoding: $\quad x \in \{-1,1\}^n \to z(x) \in \{-1,1\}^n$

modulation: $\quad z(x) \to \sum_k z_k(x)\,\delta(t - kt_0)$

transmission: $\sum_k z_k(x)\,\delta(t-kt_0) \rightarrow \sum_k z_k(x)\,h(t-kt_0)+N(t) = y(t)$

receiver: $y(t) \rightarrow \hat{x}$ best estimate of x

Under the assumption that the noise $N(t)$ is a white Gaussian noise, $\hat{x}$ is obtained by minimizing the function

$$E(x) = -\tfrac{1}{2}\int\left[\sum_k z_k(x)\,h(t-kt_0)\right]^2 dt + \int y(t)\sum_k z_k(x)\,h(t-kt_0)\,dt$$

In the uncoded case (i.e., $z_k(x)=x_k$), $E(x)$ takes on the form

$$E(x) = -\tfrac{1}{2}\sum_{i,j} W_{ij}\,x_i\,x_j + \sum_i \theta_i\,x_i$$

and (4.3) becomes

$$X_i(t+1) = \operatorname{sgn}\left[\sum_{j\neq i} w_{ij}\,X_j(t) - \tfrac{1}{2}Z(t)X_i(t) - \theta_i\right]$$

which can be implemented easily by a discrete time Hopfield net. In contrast, a conventional receiver would be made up of several stages of suboptimal operations.

For the encoded case, we need to compute $z(x)$ and $z(\overline{x_i})$. This is interesting because to compute these we need the encoder, not the decoder. Therefore, we seem to have found a way of using the encoder to decode for arbitrary encoding schemes.

6 Conclusion

The potential of neural networks to find global optimum should be further explored. Their ability to do so using only local "gradient" information is surprising and can lead to very useful applications. Implementing an optimum receiver-decoder is a particularly interesting example.

References

[HOP82] J. J. Hopfield, "Neural networks and physical systems with emerging collective computational abilities," *Proc. National Academy of Sciences* 79 (1982) 2554-2558.

[KIR83] S. Kirkpatrick, C. D. Gelatt, Jr. and M. P. Vecchi, "Optimization by simulated annealing," *Science* 220 (1983) 671-680.

[WON91a] E. Wong, "Stochastic neural networks," *Algorithmica* 6 (1991) 466-478.

[WON91b] E. Wong, "Implementing Boltzmann machines," in *Stochastic Analysis*, E. Mayer-Wolf, E. Merzbach and A. Schwartz, eds., Academic Press, San Diego, 1991.