Communicated by Scott Kirkpatrick

Deterministic Boltzmann Learning Performs Steepest Descent in Weight-Space

Geoffrey E. Hinton
Department of Computer Science, University of Toronto, 10 King's College Road, Toronto M5S 1A4, Canada

The Boltzmann machine learning procedure has been successfully applied in deterministic networks of analog units that use a mean field approximation to efficiently simulate a truly stochastic system (Peterson and Anderson 1987). This type of "deterministic Boltzmann machine" (DBM) learns much faster than the equivalent "stochastic Boltzmann machine" (SBM), but since the learning procedure for DBM's is only based on an analogy with SBM's, there is no existing proof that it performs gradient descent in any function, and it has only been justified by simulations. By using the appropriate interpretation for the way in which a DBM represents the probability of an output vector given an input vector, it is shown that the DBM performs steepest descent in the same function as the original SBM, except at rare discontinuities. A very simple way of forcing the weights to become symmetrical is also described, and this makes the DBM more biologically plausible than back-propagation (Werbos 1974; Parker 1985; Rumelhart et al. 1986).

1 Introduction

The promising results obtained by Peterson and Anderson (Peterson and Anderson 1987) using a DBM are hard to assess because they present no mathematical guarantee that the learning does gradient descent in any error function (except in the limiting case of a very large net with small random weights). It is quite conceivable that in a DBM the computed gradient might have a small systematic difference from the true gradient of the normal performance measure for each training case, and when these slightly incorrect gradients are added together over many cases their resultant might bear little relation to the resultant of the true case-wise gradients (see Fig. 1).

2 The Learning Procedure for Stochastic Boltzmann Machines

A Boltzmann machine (Hinton and Sejnowski 1986) is a network of symmetrically connected binary units that asynchronously update their states

Neural Computation **1, 143–150** (1989)

according to a *stochastic* decision rule. The units have states of 1 or 0 and the probability that unit i adopts the state 1 is given by

$$p_i = \sigma(\frac{1}{T}\sum_j s_j w_{ij}) \tag{2.1}$$

where s_j is the state of the j^{th} unit, w_{ij} is the weight on the connection between the j^{th} and the i^{th} unit, T is the "temperature" and σ is a smooth non-linear function defined as

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.2}$$

If the binary states of units are updated asynchronously and repeatedly using equation 2.1, the network will reach "thermal equilibrium" so that the relative probabilities of global configurations are determined by their energies according to the Boltzmann distribution:

$$\frac{P_\alpha}{P_\beta} = \frac{e^{-E_\alpha/T}}{e^{-E_\beta/T}} \tag{2.3}$$

where P_α is the probability of a global configuration and E_α is its energy defined by

$$E_\alpha = -\sum_{i<j} s_i^\alpha s_j^\alpha w_{ij} \tag{2.4}$$

where s_i^α is the binary state of unit i in the α^{th} global configuration, and bias terms are ignored because they can always be treated as weights on connections from a permanently active unit.

At any given temperature, T, the Boltzmann distribution is the one that minimizes the Helmholtz free energy, F, of the distribution. F is defined by the equation

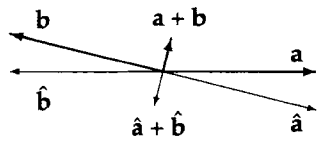

Figure 1: The true gradients of the performance measure are **a** and **b** for two training cases. Even fairly accurate estimates, **â** and **b̂**, can have a resultant that points in a very different direction.

$$F = \langle E \rangle - TH \tag{2.5}$$

where $\langle E \rangle$ is the expected value of the energy given the probability distribution over configurations and H is the entropy of the distribution. It can be shown that minima of F (which will be denoted by F^*) satisfy the equation

$$e^{-F^*/T} = \sum_{\alpha} e^{-E_\alpha/T} \tag{2.6}$$

In a stochastic Boltzmann machine, the probability of an output vector, O_β, given an input vector, I_α is represented by

$$P^-(O_\beta|I_\alpha) = \frac{e^{-F^*_{\alpha\beta}/T}}{e^{-F^*_\alpha/T}} \tag{2.7}$$

where $F^*_{\alpha\beta}$ is the minimum free energy with I_α and O_β clamped, and F^*_α is the minimum free energy with just I_α clamped. A very natural way to observe $P^-(O_\beta|I_\alpha)$ is to allow the network to reach thermal equilibrium with I_α clamped, and to observe the probability of O_β. The key to Boltzmann machine learning is the simple way in which a small change to a weight, w_{ij}, affects the free energy and hence the log probability of an output vector in a network at thermal equilibrium.

$$\frac{\partial F^*}{\partial w_{ij}} = -\langle s_i s_j \rangle \tag{2.8}$$

where $\langle s_i s_j \rangle$ is the expected value of $s_i s_j$ in the minimum free energy distribution. The simple relationship between weight changes and log probabilities of output vectors makes it easy to teach the network an input-output mapping. The network is "shown" the mapping that it is required to perform by clamping an input vector on the input units and clamping the required output vector on the output units (with the appropriate conditional probability). It is then allowed to reach thermal equilibrium at $T = 1$, and at equilibrium each connection measures how often the units it connects are simultaneously active. This is repeated for all input-output pairs so that each connection can measure $\langle s_i s_j \rangle^+$, the expected probability, averaged over all cases, that unit i and unit j are simultaneously active at thermal equilibrium when the input and output vectors are both clamped. The network must also be run in just the same way but without clamping the output units to measure $\langle s_i s_j \rangle^-$, the expected probability that both units are active at thermal equilibrium when the output vector is determined by the network. Each weight is then updated by

$$\Delta w_{ij} = \epsilon(\langle s_i s_j \rangle^+ - \langle s_i s_j \rangle^-) \tag{2.9}$$

It follows from equation 2.7 and equation 2.8 that if ϵ is sufficiently small this performs steepest descent in an information theoretic measure, G, of the difference between the behavior of the output units when they are clamped and their behavior when they are not clamped.

$$G = \sum_{\alpha,\beta} P^+(I_\alpha, O_\beta) \log \frac{P^+(O_\beta \mid I_\alpha)}{P^-(O_\beta \mid I_\alpha)} \tag{2.10}$$

where I_α is a state vector over the input units, O_β is a state vector over the output units, P^+ is a probability measured at thermal equilibrium when both the input and output units are clamped, and P^- is a probability measured when only the input units are clamped.

Stochastic Boltzmann machines learn slowly, partly because of the time required to reach thermal equilibrium and partly because the learning is driven by the *difference* between two noisy variables, so these variables must be sampled for a long time at thermal equilibrium to reduce the noise. If we could achieve the same simple relationships between log probabilities and weights in a deterministic system, learning would be much faster.

3 Mean field theory

Under certain conditions, a stochastic system can be approximated by a deterministic one by replacing the stochastic binary variables of equation 2.1 by deterministic real-valued variables that represent their mean values

$$p_i = \sigma\left(\frac{1}{T}\sum_j p_j w_{ij}\right) \tag{3.1}$$

We could now perform discrete, asynchronous updates of the p_i using equation 3.1 or we could use a synchronous, discrete time approximation of the set of differential equations

$$\frac{dp_i}{dt} = -p_i + \sigma\left(\frac{1}{T}\sum_j p_j w_{ij}\right) \tag{3.2}$$

We shall view the p_i as a representation of a probability distribution over all binary global configurations. Since many different distributions can give rise to the same mean values for the p_i we shall assume that the distribution being represented is the one that maximizes the entropy, subject to the constraints imposed on the mean values by the p_i. Equivalently, it is the distribution in which the p_i are treated as the mean values of *independent* stochastic binary variables. Using equation 2.5 we can calculate the free energy of the distribution represented by the state of a DBM (at $T = 1$).

$$F = -\sum_{i<j} p_i p_j w_{ij} + \sum_i [p_i \log(p_i) + (1 - p_i) \log(1 - p_i)] \tag{3.3}$$

Although the dynamics of the system defined by equation 3.2 do not consist in following the gradient of F, it can be shown that it always moves in a direction that has a positive cosine with the gradient of $-F$ so it settles to one of the minima of F (Hopfield 1984).

Mean field systems are normally viewed as approximations to systems that really contain higher order statistics, but they can also be viewed as exact systems that are strongly limited in the probability distributions that they can represent because they use only N real values to represent distributions over 2^N binary states. Within the limits of their representational powers, they are an efficient way of manipulating these large but constrained probability distributions.

4 Deterministic Boltzmann machine learning

In a DBM, we shall define the representation of $P^-(O_\beta|I_\alpha)$ exactly as in equation 2.7, but now $F^*_{\alpha\beta}$ and F^*_α will refer to the free energies of the particular minima that the network actually settles into. Unfortunately, in a DBM this representation is no longer equivalent to the obvious way of defining $P^-(O_\beta|I_\alpha)$ which is to clamp I_α on the input units, settle to a minimum of F_α, and interpret the values of the output units as a representation of a probability distribution over output vectors, using the maximum entropy assumption.

The reason for choosing the first definition rather than the second is this: Provided the stable states that the network settles to do not change radically when the weights are changed slightly, it can now be shown that the mean field version of the Boltzmann machine learning procedure changes each weight in proportion to the gradient of $\log P^-(O_\beta|I_\alpha)$, which is exactly what is required to perform steepest descent in the performance measure G defined in equation 2.10.

When w_{ij} is incremented by an infinitessimal amount $\epsilon p_i p_j$ two things happen to F^* (see Fig. 2). First, the mean energy of the probability distribution represented by the state of the DBM is decreased by $\epsilon p^2{}_i p^2{}_j$ and, to first order, the mean energy of all nearby states of the DBM is decreased by the same amount. Second, the values of the p_i at which F is minimized change slightly so the stable state moves slightly. But, to first order, this movement of the minimum has *no* effect on the value of F because we are at a stable state in which $\partial F/\partial p_i = 0$ for all i. Hence the effect of incrementing w_{ij} by $\epsilon p_i p_j$ is simply to create a new, nearby stable state which, to first order, has a free energy that is $\epsilon p_i^2 p_j^2$ lower than the old stable state. So, assuming $T = 1$, if all weights are incremented by $\epsilon p_i^+ p_j^+$ in the stable state that has I_α and O_β clamped and are decremented by $\epsilon p_i^- p_j^-$ in the stable state that has only I_α clamped we have, from equation 2.7

$$\Delta w_{ij} = \epsilon(p_i^+ p_j^+ - p_i^- p_j^-) = \epsilon \frac{\partial \log P^-(O_\beta | I_\alpha)}{\partial w_{ij}} \tag{4.1}$$

This ensures that by making ϵ sufficiently small the learning procedure can be made to approximate steepest descent in G arbitrarily closely.

The derivation above is invalid if, with the same boundary conditions, a small change in the weights causes the network to settle to a stable state with a very different free energy. This can happen with energy landscapes like the one shown in figure 3. A small weight change caused by some other training case can cause a free energy barrier that prevents the network finding the deeper minimum. In simulations that repeatedly sweep through a fixed set of training cases, it is easy to avoid this phenomenon by always starting the network at the stable state that was found using the same boundary conditions on the previous sweep. This has the added advantage of eliminating almost all the computation required to settle on a stable state, and thus making a settling almost as fast as a forward pass of the back-propagation procedure.

Unfortunately, starting from the previous best state does not eliminate the possibility that a small free-energy barrier will disappear and a much better state will then be found when the network is running with the output units unclamped. This can greatly increase the denominator in equation 2.7 and thus greatly decrease the network's representation of the probability of a correct output vector. It should also be noted that it is conceivable that, due to local minima in the free energy landscape, F_α^* may actually be higher than $F_{\alpha\beta}^*$, in which case the network's representation of $P^-(O_\beta | I_\alpha)$ will exceed 1. In practice this does not seem to be a problem, and DBM's compare very favorably with back-propagation in learning speed.

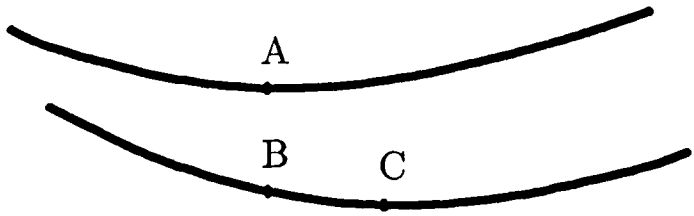

Figure 2: The effect of a small weight increment on a free energy minimum. To first order, the difference in free energy between A and C is equal to the difference between A and B. At a minimum, small changes in the distribution (sideways movements) have negligible effects on free energy, even though they may have significant (and opposite) effects on the energy and the entropy terms.

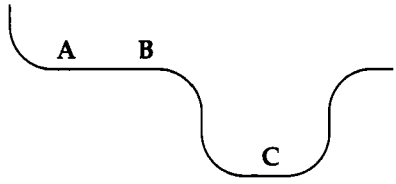

Figure 3: A small increase in the free energy of B can prevent a network from settling to the free energy minimum at C. So small changes in weights occasionally cause large changes in the final free energy.

5 Symmetry of the Weights

We have assumed that the weight of the connection from i to j is the same as the weight from j to i. If these weights are asymmetric, the learning procedure will automatically symmetrize them provided that, after each weight update, each weight is decayed slightly towards zero by an amount proportional to its magnitude. This favors "simple" networks that have small weights, and it also reduces the energy barriers that create local minima. Weight-decay always reduces the difference between w_{ij} and w_{ji}, and since the learning rule specifies weight changes that are exactly symmetrical in i and j, the two weights will always approach one another. Williams (1985) makes a similar argument about a different learning procedure. Thus the symmetry that is required to allow the network to compute its own error derivatives is easily achieved, whereas achieving symmetry between forward and backward weights in back-propagation networks requires much more complex schemes (Parker 1985).

6 Acknowledgments

I thank Christopher Longuet-Higgins, Conrad Galland, Scott Kirkpatrick, and Yann Le Cun for helpful comments. This research was supported by grants from the Ontario Information Technology Research Center, the National Science and Engineering Research Council of Canada, and DuPont. Geoffrey Hinton is a fellow of the Canadian Institute for Advanced Research.

References

Hinton, G.E. and T.J. Sejnowski. 1986. Learning and relearning in Boltzmann machines. *In:* Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations, eds. D.E. Rumelhart, J.L. McClelland, and the PDP group. Cambridge, MA: MIT Press.

Hopfield, J.J. 1984. Neurons with Graded Response Have Collective Computational Properties like Those of Two-state Neurons. *Proceedings of the National Academy of Sciences U.S.A.* **81**, 3088–3092.

Parker, D.B. 1985. *Learning-logic.* Technical Report TR-47, Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA.

Peterson, C. and J.R. Anderson. 1987. A Mean Field Theory Learning Algorithm for Neural Networks. *Complex Systems* **1**, 995–1019.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams. 1986. Learning Representations by Back-propagating Errors. *Nature* **323**, 533–536.

Werbos, P.J. 1974. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* PhD Thesis, Harvard University, Cambridge, MA.

Williams, R.J. 1985. *Feature Discovery through Error-correction Learning.* Technical Report ICS-8501, Institute for Cognitive Science, University of California, San Diego, La Jolla, CA.

Received 5 December; accepted 15 December 1988.