C1.4 Stochastic neural networks

Harold Szu and Masud Cader

Abstract

Deterministic neural networks such as backpropagation of error, multilayer perceptrons, and locally based radial basis methods have been a major focus of the neural network community in recent years. However, there has been a distinct, albeit less pronounced, interest in stochastic neural networks. In this review we provide the reader with a sense of the defining components of a stochastic neural network, as well as some of the issues arising from working with stochastic neural networks. In particular, issues revolving around hardware implementation, software simulation, and innovation are developed.

C1.4.1 Introduction

The term stochastic neural network refers to a model of computation whose output is a stochastic function of its inputs and interactions among its neurons. It primarily differs from the more popular deterministic gradient descent algorithms (e.g., *backpropagation*) in that a unit activation is not a deterministic sigmoid C1.2.3 function of the inputs, but rather a stochastic function. In addition, the learning algorithm for a stochastic machine usually implements a procedure for finding a minimum on the energy surface as well as entropy maximization (Szu 1986). Although the stochastic component increases the complexity of understanding and implementation, the reward follows from the fact that a training algorithm based on simulated annealing is, theoretically, assured to converge to the global minimum, albeit slowly.

Recent developments in stochastic neural network modeling have attempted to improve computational performance either by parallel implementation or by replacing the computation of stochastic dynamics with simpler deterministic mean field approximations (Peterson 1987, Hertz 1991, Zerubia and Rama 1993, Yuille 1994, Kappen 1995a, b), that is, estimating stochastic transitions by the mean of the transitions. The performance of such annealed estimates is addressed by Tishby (1995).

Primarily, this line of accelerating the search algorithms has been based on a deterministic Boltzmann learning procedure proposed by Hinton (1989); the 'Cauchy Machine', invented by Szu (Szu and Messner 1986, Szu 1987), which uses a Cauchy distribution to generate random flights as well as walks to new states; 'adaptive simulated annealing' models (Ingber 1995) which permit fast learning via the use of differing annealing schedules across parameter dimensions; and Markov chain Monte Carlo sampling methods for state generation (Geyer 1993). These approaches offer a faster learning procedure than the original Boltzmann machine (Hinton *et al* 1984); however, they are not without their drawbacks (Wasserman 1989a, b, Galland 1993, Ingber 1995).

Other approaches, based on the fact that sufficiently simple architectures of Boltzmann machines can learn by gradient descent on the objective function (Hopfield 1987), utilize hierarchical configurations of simple Boltzmann machines so that training may proceed by gradient descent rather than involving simulated annealing. More complex interaction is enabled through the use of configurations or Boltzmann trees (Saul and Michael 1994). Still other approaches consider the problem of training a Boltzmann machine from an information geometrical view. The alternating minimization algorithm (Byrne 1992) proposes that the learning problem be addressed by minimizing the information divergence of repeated projections of the machine states and shows the equivalence of the algorithm to gradient descent and the expectation maximization technique under specific conditions.

Further, the stochastic Helmholtz machine (Dayan *et al* 1995) illustrates an innovative statistical learning algorithm (the wake-sleep algorithm of Hinton *et al* 1995) where the stochastic neural network architecture is unsupervised. That is, a multilayer network of stochastic binary neurons is augmented by two groups of weights, a top-down generative set in addition to the bottom-up recognition set (resembling very much the biweight connectivity of the *ART* model of Carpenter and Grossberg). C2.2.1

Parberry and Schnitger (1989) augment the 'classical' Boltzmann machine model, and show that in some cases Boltzmann machines may not be much more powerful than combinatorial circuits built from Boolean threshold gates. They make a number of useful comments about the practical implementation of Boltzmann machines.

An *electronic chip implementation* of a Boltzmann machine has been developed by Alspector *et al* (1989) at Bellcore, and Skubiszewski (1992), with an *optical version* by Farhat (Farhat and Psaltis 1987, Farhat 1987). Similarly, an electronic Cauchy machine has been designed by Takefuji and Szu (1989), and its optical version realized by Scheff and Szu (1987). Recently, a Gaussian machine based on both the minimization of Helmholtz's free energy and the maximization of entropy has been studied and implemented in a chip by Akiyama *et al* (1990) at Keio University. E1.3, E1.4 E1.5

C1.4.2 Simulated annealing

Since the major ingredient in stochastic machines is the simulated annealing algorithm, we compare the Boltzmann machine and Cauchy machine in terms of different algorithms: *Boltzmann annealing* (BA) and *Cauchy annealing* (CA) in section C1.4.2. Then, we review two benchmark applications; one for finding the global minimum solution of the Traveling Salesman Problem (TSP) and a second which searches for the mini-max feature in an image processing problem in section C1.4.3.

We shall discuss the sequential algorithms used in the above parallel machine implementations as follows. In BA, a Gaussian random process is used to generate new states in the sequential algorithm. Geman and Geman (1984) have proved that the cooling schedule $T(t)$ must be inversely proportional to the logarithm of time t, in order to guarantee convergence to the global minimum. This relatively slow convergence is due to the bounded variance of the Gaussian process which constrains the neighborhood of successive samples. This bounded-variance random walk is called a local search strategy. On the other hand, if one uses an infinite-variance Cauchy random process, a faster cooling schedule that is inversely proportional to time t has been deduced by Szu (1987) in one dimension and Szu and Hartley (Szu 1987, Szu and Hartley 1987) in arbitrary higher dimensions (as applied to solving the bearing fix problem with multiple sensors and multiple targets). This new class of algorithms, implementing a semilocal search strategy, permits occasionally long steps (the so-called Lévy–Doob diffusion) far from the neighborhood of the previous sample. These random flights are indicative of the divergence of the second moment of the Cauchy probability distribution.

In a convex optimization problem, one can start at any point in the function space, measure the local gradient, and take a step in any direction which is lower in altitude than the current position. Repetition of this process will assure asymptotic convergence to the minimum (i.e., optimum) solution. In a nonconvex problem, the optimization function has multiple local minima, each with different depths, for which the optimum is defined to be the global minimum. The application of local gradient techniques to nonconvex optimization creates a problem where one becomes caught in a local minimum with no way of determining whether the local minimum is also the desired global minimum. One solution to this dilemma is to permit steps whose magnitude and direction are dependent on the local gradient and to add random noise in an annealing process Wasserman (1989a, b).

Further, for the algorithm to converge, the magnitude of the random component of the step size must decrease in a statistically monotonic fashion. In the physical annealing process these steps can be equated with Brownian motion of a particle, traveling at statistical velocity V, over an intersample time Δt. The expectation of V^2 is linearly related to the temperature of the particle. The simulated annealing community (Kirkpatrick *et al* 1983) therefore refers to the 'temperature' of the random process and uses the term 'cooling schedule' to refer to the algorithm for monotonically reducing the temperature.

An annealing methodology requires three major steps: (i) the generation of a new search state by means of a random process covering all phase space without the barrier of an energy landscape (section C1.4.2.1); (ii) the acceptance criterion of the new state, based on the energy landscape property at the new and the old states (section C1.4.2.2); and (iii) the cooling schedule for quenching the random noise used

to generate a new state together with an appropriate change in the new-state acceptance criterion (section C1.4.2.3).

C1.4.2.1 State-generating probability density

The Boltzmann machine uses a Gaussian probability density to generate the incremental displacement X between the old state x and the new state x' as follows:

$$G_T(x'|x' = x + X) = (1/2\pi T^{1/2} \exp(-X^2/T). \tag{C1.4.1}$$

Based on the central limit theorem (CLT), any random variable with a bounded variance approaches the Gaussian distribution in the large-sampling limit.

The Cauchy state generating probability density is:

$$G_T(x'|x' = x + X) = [T/\pi(T^2 + |X|^2)]. \tag{C1.4.2}$$

Both distributions can be expanded in Taylor series and become identically quadratic for small displacements. This means that locally they are both identical to random walks. However, when the second moment is taken, the Cauchy density produces an infinite divergence while the Gaussian density gives the value of the temperature. This illustrates that the Cauchy distribution will generate random flights in long steps (Lévy flights), and that the CLT does not apply.

For an optical implementation, the random displacement X can be easily generated by a uniform angle distribution between $\pm\pi/2$ by a light beam deflected from a suspended mirror on a flat screen as demonstrated previously for an optical Cauchy machine (Scheff and Szu 1987). The displacement X is measured from the center and is given by

$$X = T\tan(\theta) \tag{C1.4.3}$$

since with $\mathrm{d}\tan(\theta)/\mathrm{d}\theta = 1/(1 + \tan(\theta)^2)$, we can replace $\tan(\theta)$ with X/T yielding equation (C1.4.3).

C1.4.2.2 Local and distributed acceptance criteria

The primary difference between sequential simulations and parallel implementations of simulated annealing is that the former relies on a centralized acceptance criterion (an *uphill* energy concept), while parallel versions require a distributed criterion (an against peer pressure concept).

The total system energy is convenient for a top-down design, but is not suited for parallel implementations. Any criterion based on the total system energy requires a central processor to tally the contribution from all distributed processors. If each processor is waiting for a centralized decision, the speed of parallel execution will be slowed down.

A natural choice for a distributed acceptance criterion is one based on the interaction forces carried by local communication links. These interactions can be related to the entire energy landscape.

For example, the natural phenomenon occurring in a water–ice phase transition is a parallel and collective computation without central control where a slow cooling or annealing schedule insures the low-energy crystalline state of ice. In other words, during the occasional uphill climb of the energy landscape to detrapping (or a metastable crystalline state), there is an occasional thermal fluctuation against peer pressure. This fluctuation manifests itself via the interacting Coulomb forces which communicate instantaneously among all processors or molecules, rather than through the posterior energy landscape. A neural network is similar to this liquid–solid phase transition which promises the minimum-energy crystal state if it is cooled down properly.

If the energy change $\Delta E = E_{\mathrm{new}} - E_{\mathrm{old}}$ is less than zero, the new state is accepted. On the other hand, if the energy change ΔE is greater than zero, then the following acceptance probability is computed:

$$P_T = 1/[1 + \exp(-\Delta E/T)] \tag{C1.4.4}$$

(which is larger than a uniformly generated random number) and the uphill state is accepted, otherwise the state is rejected. Such an energy landscape formula can be thought of as a two state normalized transition probability: $\exp(-E_{\mathrm{new}})/[\exp(-E_{\mathrm{new}}) + \exp(-E_{\mathrm{old}})]$ and therefore works well on a conventional serial machine for one neuron decision at a time.

For a Gaussian noise model, the appropriate Metropolis acceptance criterion (Metropolis *et al* 1953) cannot be integrated into an elementary quadrature, which yields, by the steepest-descent approximation, the energy landscape concept ΔE. Hinton and Sejnowski have interpreted the acceptance criterion, equation (C1.4.4), as the energy change for each neuron, ΔE_i, which is used to derive a specific hidden layer weight, in order to derive a local acceptance criterion (cf see appendix of Hinton and Sejnowski (1986)).

A one-dimensional optically implemented neural network utilizing CA has been developed as the Cauchy Machine (Scheff and Szu 1987). However, a local distributed VLSI design could not be implemented until a distributed acceptance criterion was derived for the Cauchy density (Takefuji and Szu 1989).

If the total input u_i to the McCulloch–Pitts model of a binary neuron is defined as

$$u_i = \sum_j T_{ij} v_j$$

then, as consistent with the Metropolis acceptance criterion, the output v_i is locally set to be one only if random numbers generated within the interval [0, 1] are less than the acceptance function—which is integrated exactly for each total input as follows:

$$(1/\pi T) \int_0^\infty \mathrm{d}x/[1 + ((x - u_i)/T)^2] = (1/2) + \tan^{-1}(u_i/T(t))/\pi. \tag{C1.4.5}$$

In the case of annealing, the inverse of the cooling schedule is defined to be the piecewise constant gain coefficient, G_n, at a positive integer time point t_n:

$$G(t_n) = 1/T(t_n) = G_n. \tag{C1.4.6}$$

Then, the output v_i also fluctuates within a finite bound described as both firing rate transfer functions:

$$v_i = \sigma_{1n}(u_i) = (1/2) + \tan^{-1}(u_i G_n)/\pi. \tag{C1.4.7}$$

Note that equation (C1.4.7) is almost identical to the standard sigmoidal/logistic function of $1/[1 + \exp(-u_i G_n)]$, except that the arctangent function becomes slightly rounded near the central region. In the case of the sigmoidal function, the slope σ_n' is proportional to the gain coefficient G_n:

$$\sigma_n' = \mathrm{d}v_i/\mathrm{d}u_i = G_n v_i(1 - v_i). \tag{C1.4.8}$$

When $T = 0$, the infinite gain G implies an infinite slope. In this limit, both firing rate transfer functions become a binary step function $v_i = \mathrm{step}(u_i)$ describing a binary neuron model. Thus, the annealing process gradually changes a sigmoidal neuron toward a binary neuron.

C1.4.2.3 Annealing cooling schedules

The cooling schedule is critical to the performance of the learning algorithm. For a given random process, cooling at too fast a rate will probably 'freeze' the system in a nonglobal minimum. Cooling at too slow a rate, while reaching the desired global minimum, is a waste of computational resources. The technical problem is to derive the fastest cooling schedule that will guarantee convergence to the global minimum. With this understanding, the term 'cooling schedule' is synonymous with 'permissible fastest cooling schedule' during which the complete phase space is guaranteed to be available for searching at all time.

Without any knowledge of energy landscapes, one can only hope to derive an appropriate cooling schedule for a specific stochastic process. The necessary condition is that at any temperature the phase space is always accessible infinitely often in time (IOT). In other words, an inappropriately fast cooling schedule may quench the IOT availability of some remote states, and hence, not find the global minimum. The specific energy landscape and an appropriate acceptance criterion must be taken into consideration to determine whether the minimum will be actually be found. Ingber (1993), has shown that exponential cooling schedules may be used but only with specific distributional forms used as the state generating function.

For a Gaussian random process, Geman and Geman (1984) have proved that the simulated annealing cooling schedule of the temperature $T(t)$ must be decreased (from a given sufficiently high temperature T_0 down to zero 'degrees') according to the inverse logarithmic formula:

$$T = T_0/\log(1 + t). \tag{C1.4.9}$$

Thus, in the interest of speeding up the annealing process and yet maintaining the capability of finding the global minimum, Szu *et al* applied Cauchy colored noise to the problem, instead of a Gaussian random process. The resultant cooling schedule for an arbitrary initial temperature is derived:

$$T = T_0/(1+t) \tag{C1.4.10}$$

which is indeed faster, and was shown to insure that the complete search space is available at all temperatures.

The mathematical truth in both proofs is based on the fact that the infinite series of the inverse time steps is divergent from an arbitrary initial time point t_0

$$\sum_{t=t_0}^{\infty} \frac{1}{t} = \infty. \tag{C1.4.11}$$

The complete proofs for both are provided in appendix A.

It is useful to note that CA is $t/\log(t)$ faster than a Gaussian (white noise) simulated annealing algorithm which in turn is superior to the conventional Monte Carlo method in which the temperature is held constant.

C1.4.3 Applications

A stochastic neural network model, the Boltzmann machine, has been used in demonstrating the celebrated Net-Talk (Sejnowski and Rosenberg 1987). Similarly, the problem of obtaining rapid and accurate estimations of the locations of moving emitters from samples of imprecise bearing only data has been addressed with the Cauchy machine (Szu 1987). Another innovative application has been the use a class of BM (in which visible units are connected only to hidden units) to repair a dataset with missing values (Kappen 1995a, b).

In the remainder of this section, we illustrate two applications of the Cauchy machine. The first, a benchmark problem in this area, is that of determining the shortest tour length of a traveling salesman through a set of cities only taking into account the constraint of distance between cities. The second is a problem related to optical character recognition, where the idea is to automatically extract features from the character pattern sets.

C1.4.3.1 Constraint specifications

A traveling salesman problem (TSP), which attempts to find the shortest possible tour through a given number of cities, can be stochastically solved by generating noise via the leptokurtic Cauchy probability density, $T/\pi(T^2+X^2)$ (Szu 1990). The noise must be quenched with the inversely linear cooling schedule: $T = T_0/(1+t)$ as described earlier. Moreover, the schedule must be followed consistently for every time step, both in generating new states and in visiting some of the states whenever the acceptance criterion is met.

The performance of CA was calibrated by comparing against the results obtained by an exhaustive search through all possible TSP solutions. This is possible due to a novel factorial number representation for each TSP configuration by an integer n described as follows.

We require a one-dimensional coding scheme for the TSP search space that is one-to-one unique. Due to the combinatorial nature of the TSP, a good guess at a number representation might be a factorial number base system. We adopt, in the following manner, a coding scheme as follows:

(i) The real line x is sampled by the set of real integers x, using the function Int( ).
(ii) Then, integers are made periodically in the modulus base set of $(N-1)!$, using the Mod(,) function.
(iii) Such an integer number represents a state of a valid tour since a factorial base set is related to the tour order permutations. Thus, one represents the integer in terms of the factorial number base system by calculating the most significant numbers denoted by the index tuple,

$$X_{\text{new}} = \sum_n \text{index}_n \times n! \tag{C1.4.12a}$$

$$X_{\text{new}} \leftrightarrow (\text{index}_{N-1}, \text{index}_{N-2}, \ldots, \text{index}_1, \text{index}_0) \tag{C1.4.12b}$$

sequentially for all n beginning with $N-1$ down to 0.

To produce the set of indices, one considers an example for five cities, $N = 5$, denoted by city: Nos 1–5. Given $X_{\text{old}} = 0 =$ (No 1, No 2, No 3, No 4, No 5) as a reference (the diagonal matrix element of Hopfield–Tank), where the arbitrary tour order is that city No 1 is visited first, and so on. One finds,

$$X_{\text{new}} = 15 = 0 \times 0! + 1 \times 1! + 1 \times 2! + 2 \times 3! + 0 \times 4! \leftrightarrow (\text{No 1, No 4, No 3, No 5, No 2})$$

where the representation index $= (0, 1, 1, 2, 0)$ is obtained with respect to the base set (0!, 1!, 2!, 3!, 4!).

The energy corresponding to each of the possible round-trip routes through n cities, $4 \leq n \leq 10$, has been reported (Szu and Scheff 1990), so, while the exhaustive search through hundreds of thousand of possible cases used several hours of computer time on a Mac II (with factorial scaling implying that five hours for ten cities would require 50 hours for 11 cities). In contrast, CA took about 10 minutes or less to find the global minima for the ten-city problem. As the shortest tours agreed with those found by CA, it is clear that CA is superior because the search required a sampling of less than 1% of the states, with another 2% sampling to verify the stability. Thus, it is evident that traditional Monte Carlo random sampling should be replaced with the CA algorithm.

C1.4.3.2 Image processing and pattern recognition

F1.2, F1.6

Geman and Geman (1984) have applied Boltzmann annealing to the problem of noisy image restoration. Smith *et al* (1983) have also applied BA to radiology image reconstructions. Szu and Scheff (1990) have shown that CA can also be useful in pattern recognition. In particular, they have used a minimax cost function to investigate the self-extraction of unkown features, previously accomplished using self-reference matched filters (Szu *et al* 1980, Szu and Blodgett 1982, Szu and Messner 1986).

Let the critical feature of the template class c be denoted as $f_c(x, y)$. Then, a space-filling curve, Peano N-curve, is employed to replace the traditional line-by-line scan sampling, in order to preserve the neighborhood proximity relationship.

The performance criterion seeks to minimize the distance between the image template I_c of the c-class ($c = 1, 2$), to minimize the inner product between classes $\langle f_c | f_{c'} \rangle$, and to maximize the distance $|f_c - f_{c'}|^2$ between two feature vectors. Thus, the *minimax* energy for the determination of the global minimum associated with the unknown feature f_c is given by,

$$E(f_c) = a \sum_{c \neq c'} (\langle f_c | f_{c'} \rangle) + b \sum_{c=1,2} |f_c - I_c|^2 + d \sum_{c \neq c'} 1/|f_c - f_{c'}|^2. \tag{C1.4.13}$$

Note how the representation permits parametrization of relative feature importance. For example, the Lagrangian multipliers $a = 10$ and $d = 10$ are set higher than $b = 1$ to reflect the less important fact that feature f_c should resemble image I_c. The results using the CA algorithm are provided by Szu (1990). A sample listing in a variant of Basic is provided in appendix B.

In these examples, we have focused on representation issues which clearly have significant impact on the performance of the algorithms. There is nothing significantly unique about the need for representation encodings in neural network applications; however, in digital simulations of stochastic neural models any time improvement afforded by clever representation greatly facilitates the application.

C1.4.4 Summary

We have illustrated that Cauchy annealing is superior to Boltzmann annealing, which in turn is superior to conventional Markov Monte Carlo methods. We have illustrated or referenced how digital implementations of stochastic neural networks are inefficient, except, perhaps, when coupled with mitigating factors such as clever problem representation, deterministic annealing, adaptive simulated annealing, or composite hierarchical architectural topologies (trees for example). It is also clear that for hard problems of large scale, analog (especially optical) implementations hold great promise for extending the applicability of stochastic neural networks.

Appendix A. Proofs of both cooling schedules

There are a number of similarities in the proofs of the cooling schedules for the CA and BA algorithms in D-dimensional vector spaces. For the convenience of comparison, the proofs will be demonstrated in

parallel. In locating the minimum, one must start at some position or state in a D-dimensional space, evaluate the function at that state, and generate the next state vector.

The CA and BA algorithms are different in that CA uses a Cauchy distribution and BA uses a Gaussian distribution in their respective state generating functions. Both the BA and CA algorithms will use as their next state either the current state vector or the next state vector provided its incremental cost increase is less than the time-dependent noise bound, which is temperature (and therefore time) dependent.

The CA algorithm requires that *state generating* be *infinitely often in time (IOT)* (in the sense of accumulation in time defined by the negation below) whereas the BA requires the *state visiting* be IOT. At some cooling temperature $T_c(t)$ at time t, let the state generating probability of being within a specific neighborhood be lower bounded by g_t. Then the probability of not generating a state in that neighborhood is upper bounded by $(1 - g_t)$. To insure a globally optimum solution for all temperatures, a state in an arbitrary neighborhood must be able to be generated IOT, which however does not imply ergodicity, the latter requiring actual visits IOT. To prove that a specific cooling schedule maintains the state generation IOT, it is easier to prove the *negation* of the *converse*, namely the *impossibility of never generating a state in the neighborhood after an arbitrary time* t_0. Mathematically this is equivalent to stating that the infinite product of $|1 - g_t|$ terms is zero. Taking the Taylor series expansion of the logarithm of the product, one can alternatively prove that the sum of the g_t terms is infinite. One can now verify cooling schedules in a D-dimensional neighborhood $|\Delta x_0|$ and arbitrary time t_0. Among the various Lévy–Doob distributions (including Cauchy, Holtzmach, and Gaussian) there are two different classes, local (as in CA) and semilocal (as in CA). There exists an initial temperature T_0 and for $t > 0$, such that

$$\text{BA} : T_a(t) = T_0/\log(t) \tag{C1.4.A1a}$$

$$\text{CA} : T_c(t) = T_0/t \tag{C1.4.A1b}$$

$$\text{BA} : g_t \approx \exp\left(\frac{|\Delta x_0^2|}{-T_a(t)}\right) T_a(t)^{D/2} \tag{C1.4.A2a}$$

$$\text{CA} : g_t \approx \frac{T_c(t)}{\left(T_c^2 - |\Delta x_0|^2\right)^{(D+1)/2}} \approx \frac{T_0}{t|\Delta x_0|^{D+1}} \tag{C1.4.A2b}$$

$$\text{BA} : \sum_{t=t_0}^{\infty} g_t \geq \sum_{t=t_0}^{\infty} \exp(-\log(t)) = \sum_{t=t_0}^{\infty} \frac{1}{t} = \infty \tag{C1.4.A3a}$$

$$\text{CA} : \sum_{t=t_0}^{\infty} g_t \approx \frac{T_0}{|\Delta x_0|^{D+1}} \sum_{t=t_0}^{\infty} \frac{1}{t} = \infty. \tag{C1.4.A3b}$$

Appendix B. Cauchy annealing algorithm (Macintosh QuickBasic version)

```
! input two known images and known feature
DATA 4,5,8,9,11,14,15,16,17,38,41,44,46,47,50,51,52,53,56,57 !input 81 Peano-scanning pixel#
        for the black value=1
DATA 58,59,67,69,70,71,72,78,79        !1= black gun barrel, track belt
DATA 4,5,8,9,12,13,14,15,16,17,30,31,37,42,43,46,47,50,51,52
DATA 53,56,57,58,59,62,63,69,70
DIM f1(81),f2(81),ave1(81),ave2(81),ft1(81),ft2(81)
MAT ave2 =0        ! True_Basic Matrix Equating
FOR n=1 to 29        ! read the tank into ave1, namely I1
    READ k
    LET ave1(k)=1
NEXT n
FOR m = 30 to 58        ! read the carrier into ave2, namely I2
READ J
LET ave2(J)=1
NEXT m
! set up Cauchy annealing to determine the unknown feature by mini-max
RANDOM        !random number rnd generated [0,1]
FOR t=1 to tmax        !after initialize the display-
```

```
    LET temp=To/(1+t)        !Cauchy annealing cooling schedule
    LET theta=(rnd-.5)*Pi        !uniform theta using the radian angle option
    LET dx=int(temp*tan(theta))        ! new pixel by T tan(theta)
    LET xnew=mod(x+dx,82)        ! modulo for 81 scan pixels
    IF xnew=0 then LET xnew=81
    IF f2(xnew)=0 THEN
        LET ft2(xnew)=ave2(xnew)
        LET ft1(xnew)=0
    ELSE
        LET ft2(xnew)=0
        LET ft1(xnew)=ave1(xnew)
    END IF
    LET enew= 0
    LET denominator=0
    LET ef1=0
    LET ef2=0
    FOR n=1 to 81
        LET ef1=ef1+(ft1(n)-ave1(n))*(ft1(n)-ave1(n))
        LET ef2=ef2+(ft2(n)-ave2(n))*(ft2(n)-ave2(n))
        LET denominator=denominator+(ft1(n)-ft2(n))*(ft1(n)-ft2(n))
        LET enew = enew + ft1(n)*ft2(n)
    NEXT n
    LET enew= a*enew + b*ef1 + c*ef2 + (d/denominator)
    IF enew<eold then
        MAT f2=ft2
        MAT f1=ft1
        LET eold=enew
        LET x=xnew
    END IF
    IF enew>=eold then
        IF(rnd*0.5)< (1/(1 + exp((enew-eold)/temp))) then        !up-hill climbing
          MAT f2=ft2
          MAT f1=ft1
          LET eold=enew
          LET x=xnew
        END IF
    END IF
! plotting search states, accepted states, and its minimax energy value
    PLOT POINTS :t,xnew+200
    PLOT POINTS :t,x+100
    PLOT POINTS :t,eold/2
```

References

Akiyama Y Y, Anzai Y and Aiso H 1990 The Gaussian machine: a stochastic, continuous neural network model *J. Neural Network Comput.* **2** (3) 43–51

Alspector J, Guputa B and Allen R 1989 Performance of stochastic learning microchip *Neural Information Processing Systems I* (Morgan Kaufmann)

Byrne W 1992 Alternating minimization and Boltzmann machine learning *IEEE Trans. Neural Networks* **3** 612–20

Dayan P, Hinton G E and Neal R M 1995 The Helmholtz machine *Neural Comput.* **7** 889–904

Farhat N H 1987 Optoelectronic analogs of self-programming neural nets: Architecture and methodologies for implementing fast stochastic learning by simulated annealing *Appl. Opt.* **26** 5093–103

Farhat N H and Psaltis D 1987 Optical implementation of associative memory based on models of neural networks *Optical Signal Processing* ed J L Horner (New York: Academic)

Galland C C 1993 The limitations of deterministic Boltzmann machine learning *Network: Computational Neural Syst.* **4** 355–79

Geman S and Geman D 1984 Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images *IEEE Trans. Pattern Anal. Machine Intell.* **6** 614–34

Geyer C J 1993 *Annealing Markov Chain Monte Carlo with Applications to Ancestral Inference* University of Minnesota

Hertz J A P, Krogh R G and Anders S 1991 *Introduction to the Theory of Neural Computation* (Reding, MA: Addison-Wesley)

Hinton G E 1989 Deterministic Boltzmann learning performs most steep descent is weight space. *Neural Comput.* **1** 143–50

Hinton G E, Dayan P, Frey B J and Neal R N 1995 *The Wake–Sleep Algorithm for Unsupervised Neural Networks* University of Toronto

Hinton G E and Sejnowski T J 1986 Learning and Relearning in Boltzmann Machines *Parallel Distributed Processing* ed J Clelland and D Rumelhart (Cambridge, MA: MIT Press) pp 282–317

Hinton G E, Sejnowski T J and Ackley D H 1984 *Boltzmann Machines: Constrained Satisfaction Networks that Learn* Carnegie Mellon University

Hopfield J J 1987 Learning algorithms and probability distributions in feed-forward and feed-back networks *Proc. Natl Acad. Sci. USA* **84** 8429–33

Ingber L 1993 Simulated annealing: Practice versus theory *Math. Comp. Modeling* **18** 29–57

——1995 Adaptive simulated annealing (ASA): lessons learned *Control and Cybernetics Preprint*

Kappen H J 1995a Deterministic learning rules for Boltzmann Machines. *Neural Networks* **8** 537–548

——1995b *Radial basis Boltzmann machines and learning with missing values* University of Nijmegen

Kirkpatrick S, Gelatt C Jr and Vecchi M P 1983 Optimization by simulated annealing *Science* **220** 671–80

Metropolis N, Rosenbluth A W, Rosenbluth M N and Teller A H 1953 Equations of state calculations for fast computing machines *J. Chem. Phys.* **21** 1087–91

Parberry I and Schnitger G 1989 Relating Boltzmann machines to conventional models of computation. *Neural Networks* **2** 29–67

Peterson C 1987 A mean field theory learning algorithm for neural networks *Complex Syst.* **1** 995–1019

Saul L J and Michael I 1994 Learning in Boltzmann trees *Neural Comput.* **6** 1174–84

Scheff K and Szu H 1987 1-D optical Cauchy machine infinite film spectrum search *IEEE Int. Conf. on Neural Networks (San Diego, 1987)*

Sejnowski T J and Rosenberg C R 1987 Parallel networks that learn to pronounce English text *Complex Syst.* **1** 145–68

Skubiszewski M 1992 *An Exact Hardware Implementation of the Boltzmann Machine* Digital Equipment Corporation

Smith W E, Barrett H H and Paxman R G 1983 Reconstruction of objects from coded images by simulated annealing *Opt. Lett.* **8** 199–201

Szu H H 1986 Non-convex optimization. Real time signal processing IX *SPIE* vol 698 (Bellingham, WA: SPIE) pp 59–65

——1987 *Fast simulated annealing* Neural Networks for Computing, Snow Bird, Utah (New York: AIP)

Szu H and Blodgett J 1982 Self-reference spatiotemporal image–restoration technique. *J. Opt. Soc. Am.* **72** 1666–9

Szu H, Blodgett J and Sica L 1980 Local instances of good seeing *Opt. Commun.* **35** 317–22

Szu H and Hartley R 1987 Nonconvex optimization by fast simulated annealing *Proc. IEEE* **75** 1538–40

Szu H and Messner R 1986 Adaptive invariant novelty filters *Proc. IEEE* **74** 519

Szu H and Scheff K 1990 Simulated annealing feature extraction from occluded and cluttered objects *Int. Joint Conf. on Neural Networks (Washington, DC, 1990)*

Takefuji Y and Szu H 1989 Parallel distributed cauchy machine *Int. Joint Conf. on Neural Networks (Washington, DC, 1990)*

Tishby N 1995 Statistical physics models of supervised learning *The Mathematics of Generalization. Proc. SFI/CNLS Workshop on Formal Approaches to Supervised Learning* (Santa Fe, NM: Addison-Wesley)

Wasserman P D 1989a A combined back-propagation/Cauchy machine network *J. Neural Network Comput.* **1** (3) 34–40

——1989b *Neural Computing Theory and Practice* (New York: Van Nostrand Reinhold)

Yuille A L 1994 Statistical Physics Algorithms that Converge. *Neural Comput.* **6** 341–56

Zerubia J and Rama C 1993 Mean field annealing using compound Gauss–Markov random fields for edge detection and image estimation *IEEE Trans. Neural Networks* **4** 703–9