NEURAL NETS, HOPFIELD

The development of artificial neural networks has been motivated by the desire to find improved methods of solving problems that are difficult for traditional computing software or hardware. The success of early neural networks led to the claim that they could solve virtually any type of problem. Although this was quickly shown to be overly optimistic, research continued during the 1970s into the use of neural networks, especially for pattern association problems. The early 1980s marked the beginning of renewed widespread interest in neural networks. A key player in the increased visibility of, and respect for, neural networks is physicist John Hopfield of the California Institute of Technology. Together with David Tank of AT&T Laboratories, Hopfield developed a group of recurrent networks that are known as Hopfield neural networks (HNN). The first of these, the discrete Hopfield neural network (DHNN), was designed as a content addressable memory (CAM). The continuous Hopfield neural network (CHNN) can also serve as a CAM, but is most widely used for combinatorial optimization problems.

One of the reasons that Hopfield's work caught the attention of the scientific community and the public was the close connection between the models and the successful development of neural network chips by researchers at AT&T and by Carver Mead and coworkers. Hopfield's emphasis on practical implications made the engineering connection very strong. By making explicit the relationship between the HNN and electrical circuits, Hopfield opened the field of neural networks to an influx of physical theory. Although many of the concepts incorporated in the HNN had antecedents in earlier neural network research, Hopfield and Tank brought them together with both clear mathematical analysis and strong emphasis on practical applications (1).

ARTIFICIAL NEURAL NETWORKS

An artificial neural network (ANN) approach to problem solving is inspired by certain aspects of biological nervous systems. An ANN is composed of a large number of very simple processing elements (neurons). The neurons are interconnected by weighted pathways. The pattern of connection among the neurons is called the network architecture. At any time, a neuron has a level of activity, which it communicates to other neurons by sending it as a signal over these pathways. Since the weights on the pathways contain much of the important information in the network, the information is distributed, rather than localized as in traditional computers.

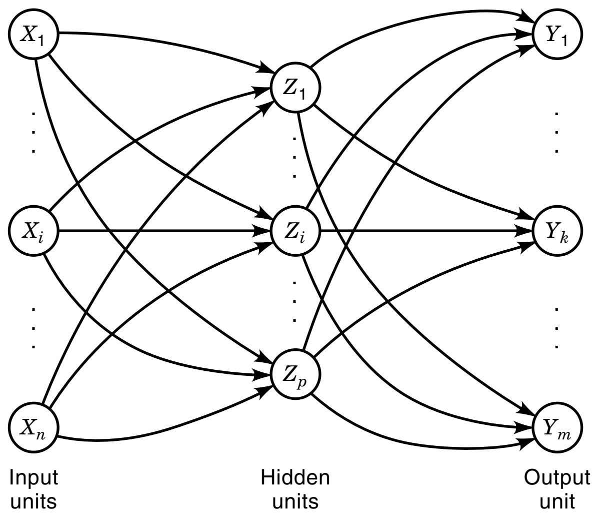

Figure 1. A feedforward network allows signals to flow from the input neurons ($X_1, \ldots, X_n$) through the hidden layer ($Z_1, \ldots, Z_p$) to the output layer ($Y_1, \ldots, Y_m$).

Architectures

One of the most basic distinctions between different types of neural networks is based on whether the network architecture allows for feedback among the neurons. A simple layered, or feed-forward, network is illustrated in Fig. 1. A fully interconnected recurrent network is shown in Fig. 2.

Weights

In addition to the design of the ANN architecture, a major consideration in developing a neural network is the determination of the connection weights. For many networks this is done by means of a training phase, in which known examples of the desired input–output patterns are presented to the network and the weights are adjusted according to a specified training algorithm. This is especially typical of feed-forward networks. In the standard Hopfield networks, the weights are fixed when the network is designed.

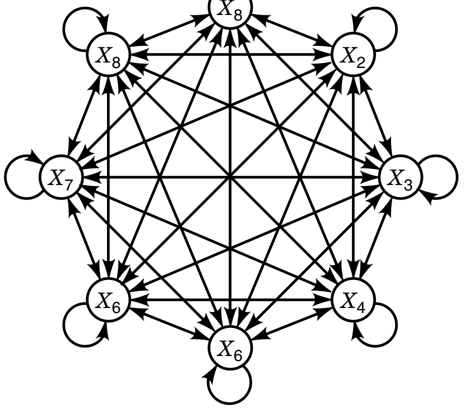

Figure 2. A fully interconnected network allows signals to flow between neurons.

Network Operation

To use a neural network, after the weights are set, an input pattern is presented and the output signal of each neuron is adjusted according to the standard process for the specific ANN model. In general, each neuron sends its output signal to the other neurons to which it is connected; the signal is multiplied by the weight on the connection pathway; each neuron sums its incoming signals. Each neuron's output signal is a nonlinear function of its summed input. In a feedforward network these computations are performed one layer at a time, starting with the input units, and progressing through the network to the output units. For a recurrent network, such as an HNN, the updating of each neuron's activity level continues until the state of the network (the pattern of activations) converges. The process differs for the discrete and continuous forms of HNN; before discussing the details, we summarize the primary types of applications for which Hopfield neural networks are used.

APPLICATIONS OF RECURRENT NEURAL NETWORKS

Memory in biological systems is fundamentally different than in a traditional digital computer, in which information is stored by assigning an address, corresponding to a physical location, where the data are written. On the other hand, your memory of an event is a combination of many sights, sounds, smells, etc. The idea of associative memory came from psychology rather than engineering, but during the 1970s much of the neural network research (especially work by James A. Anderson at Brown University and Teuvo Kohonon at the University of Helsinki) focused on the development of mathematical models of associative (or content addressable) memory. The use of an energy function analysis facilitates the understanding of associative memories that can be constructed as electronic "collective-decision circuits" (2).

The process used by biological systems to solve optimization problems also differs from that used in traditional computing techniques. Although no claim is made that neural network approaches to optimization problems directly model the methods used by biological systems, ANNs do have some potential advantages over traditional techniques for certain types of optimization problems. ANNs can find near-optimal solutions quickly for large problems. They can also handle situations in which some conditions are desirable but not absolutely required. Neural network solutions (and in particular HNN) have been investigated for many applications because of their potential for parallel computation and computational advantage when they are implemented with analog VLSI techniques.

Many other forms of recurrent neural networks have also been developed. Networks with specific recurrent structure are used for problems in which the signal varies with time. Neural networks for the study of learning, perception, development, cognition, and motor control also utilize recurrent structures.

Associative Memory

One important use of an HNN is as an autoassociative memory, which can store (or memorize) a certain number of patterns. When a modified form of one of the stored patterns is presented as input, the HNN is able to recall the original pattern after a few iterations.

Before the weights of an associative memory neural network are determined, the patterns to be stored must be converted to an appropriate representation for computation. Usually each pattern is represented as a vector with components that are either 0 or 1 (binary form) or ± 1 (bipolar form); the bipolar form is often computationally preferable for associative memory applications. The same representation is also used for patterns that are presented to the network for recognition.

Optimization

The second primary area of application for Hopfield neural networks is combinatorial optimization problems. The use of a continuous HNN for solving optimization problems was first illustrated for the traveling salesman problem (TSP), a well-known but difficult optimization problem (3) and a task assignment problem (2). Since then, HNN have been applied to optimization problems from many areas, including game theory, computer science, graph theory, molecular biology, VLSI computer-aided design, reliability, and management science. Many examples are included in Ref. 4.

The HNN approach is based on the idea that the network weights and other parameters can be found from an energy function; the network configuration (pattern of neuron activations) that produces a minimum of the energy function corresponds to the desired solution of the optimization problem. The appropriate choice of energy function for a particular problem has been the subject of much research.

DISCRETE HOPFIELD NETWORKS

The iterative autoassociative network developed by Hopfield (5,6) is a fully interconnected neural network, with symmetric weights and no self-connections, that is, $w_{ij} = w_{ji}$ and $w_{ii} = 0$. In a discrete Hopfield neural network (DHNN) only one unit updates its activation at a time (this update is based on the signals it receives from the other units). The asynchronous updating of the units allows an energy (or Lyapunov) function to be found for the network. The existence of such a function forms the basis for a proof that the network will converge to a stable set of activations. Nonsymmetric weights can lead to an unstable network.

Operation

Setting the Weights. The earliest version of the DHNN used binary input vectors; later descriptions are often based on bipolar inputs. The weight matrix to store a pattern, represented as the column vector, $\boldsymbol{p} = (p_1, \ldots, p_i, \ldots, p_n)^{\mathrm{T}}$ is the matrix $\boldsymbol{p}\boldsymbol{p}^{\mathrm{T}} - \mathrm{I}$. The matrix $\boldsymbol{p}\boldsymbol{p}^{\mathrm{T}}$ is known as the outer or matrix product of the vectors $\boldsymbol{p}$ and $\boldsymbol{p}^{\mathrm{T}}$, in contrast to the inner or scalar product $\boldsymbol{p}^{\mathrm{T}}\boldsymbol{p}$. Subtracting the identity matrix has the effect of setting the diagonal entries to 0, which is necessary to allow the network to reconstruct one of the stored patterns when a degraded or noisy form of the pattern is presented as input. For example, to find the weight matrix to store the pattern $p = (1,-1,1)^T$, we first compute the outer product

$$\boldsymbol{pp}^T = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} [1 \quad -1 \quad 1] = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix}.$$

Then

$$W = \boldsymbol{pp}^T - \mathrm{I} = \begin{bmatrix} 0 & -1 & 1 \\ -1 & 0 & -1 \\ -1 & -1 & 0 \end{bmatrix}.$$

The weight matrix W in which several patterns are stored is the sum of the individual matrices generated for each pattern.

Updating the Activations. To use a DHNN to recall a stored pattern, an input stimulus pattern $\boldsymbol{x}$ is presented to the network (one component to each neuron). Typically the input is similar to one of the stored memories. Each neuron transmits its signal to all of the other neurons. The signal received by the ith neuron is $\Sigma_j x_j w_{ji}$; by the symmetry of the weights, this is also the ith row of the product $W\boldsymbol{x}$. One neuron, chosen at random, updates its activation. Its activation is 1 if the signal it received was non-negative, that is, if $\Sigma_j x_j w_{ji} \geq 0$; the activation is -1 if $\Sigma_j x_j w_{ji} < 0$. The new pattern is again broadcast to all neurons, and another neuron is chosen to update its activation. The process continues until the network reaches a stable state, a configuration of activations that does not change.

Example. To illustrate the use of a DHNN, consider the following simple example, adapted from Ref. 7. Suppose we wish to store the three bipolar patterns

$$\begin{aligned} p_1 &= (\ \ 1 \quad\ \ 1 \quad\ \ 1 \quad\ \ 1 \quad\ \ 1)^T \\ p_2 &= (\ \ 1 \quad -1 \quad -1 \quad\ \ 1 \quad -1)^T \\ p_3 &= (-1 \quad\ \ 1 \quad -1 \quad -1 \quad -1)^T \end{aligned}$$

The weight matrix to store these three patterns is $W_1 + W_2 + W_3 = W$:

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & -1 & 1 & -1 \\ -1 & 0 & 1 & -1 & 1 \\ -1 & 1 & 0 & -1 & 1 \\ 1 & -1 & -1 & 0 & -1 \\ -1 & 1 & 1 & -1 & 0 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & -1 & 1 & 1 & 1 \\ -1 & 0 & -1 & -1 & -1 \\ 1 & -1 & 0 & 1 & 1 \\ 1 & -1 & 1 & 0 & 1 \\ 1 & -1 & 1 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -1 & 1 & 3 & 1 \\ -1 & 0 & 1 & -1 & 1 \\ 1 & 1 & 0 & 1 & 3 \\ 3 & -1 & 1 & 0 & 1 \\ 1 & 1 & 3 & 1 & 0 \end{bmatrix}$$

We present $\boldsymbol{x} = (1,-1,-1,1,1)^T$ as an input (or probe) vector, which differs from the second stored pattern in only the last component. To update the network, compute $W\boldsymbol{x} = (4,-3,4,4,-2)^T$. If the third neuron is chosen, its activation will change from -1 to 1, since it received a signal of 4. Using the updated vector of activations, $(1,-1,1,1,1)^T$ gives $W\boldsymbol{x} = (6,0,4,6,4)^T$. If neuron 1, 3, 4, or 5 is chosen, its activity will not change, and eventually neuron 2 will be chosen. Since we are using the convention that a neuron's activity is set to 1 if it receives a non-negative signal, neuron 2 will change its activation and the updated vector of activations becomes $(1,1,1,1,1)^T$, which is the first stored pattern but not the stored pattern that is most similar to the probe. If the fifth neuron had been chosen for the first update (instead of the third neuron) the network would have reached the second stored pattern immediately.

This example illustrates both the operation of a discrete Hopfield network for use as an associative memory and some of the issues that must be considered. These include questions concerning the circumstances under which convergence to a stable state is guaranteed, the question as to whether that stable state will be one of the stored patterns (and if so, will it be the closest pattern to the input?), and the relationship between the number of stored memories and the ability of the network to recall the patterns with little or no error.

Issues

Convergence. For any iterative process it is important to understand its convergence characteristics. It can be shown that the general DHNN will converge to a stable limit point (pattern of activation of the units) by considering an energy function for the system. An energy function is a function that is bounded below and is a nonincreasing function of the state of the system. For a neural network, the state of the system is the vector of activations of the units. Thus, if an energy function can be found for an iterative neural network, the ANN will converge to a stable set of activations.

The general DHNN allows an external signal y_i to be maintained during processing, so that the total signal received by neuron X_i is $y_i + \Sigma_j x_j w_{ji}$. The threshold for determining whether a neuron is "on" or "off" may be set to any desired constant θ_i; when chosen to update its activation, a unit will set its activation to "on" if

$$y_i + \sum_j x_j w_{ji} \geq \theta_i$$

and a unit will set its activation to "off" if

$$y_i + \sum_j x_j w_{ji} < \theta_i$$

An energy function for the general DHNN described here, is given by

$$E = -0.5 \sum_{i \neq j} \sum_j x_i x_j w_{ij} - \sum_i x_i y_i + \sum_i \theta_i x_i \qquad (1)$$

If the activation of the net changes by an amount Δx_i, the energy changes by the corresponding amount

$$\Delta E = -\left(y_i + \sum_{i \neq j} x_j w_{ij} - \theta_i\right) \Delta x_i \tag{2}$$

To show that $\Delta E \leq 0$, consider the two cases in which the activation of neuron X_i will change.

1. If X_i is on, it will turn off if $y_i + \sum_j x_j w_{ji} < \theta_i$. This gives a negative change for x_i. Since the quantity $y_i + \sum_{i \neq j} x_j w_{ij} - \theta_i$ in the expression for ΔE is also negative, we have $\Delta E < 0$.
2. On the other hand, if X_i is off, it will turn on if $y_i + \sum_j x_j w_{ji} > \theta_i$. This gives a positive change for x_i. Since $y_i + \sum_{i \neq j} x_j w_{ij} - \theta_i$ is positive in this case, the result is again that $\Delta E < 0$.

Therefore, the energy cannot increase. Since the energy is bounded, the network must reach a stable equilibrium where the energy does not change with further iteration. This proof uses the fact that the energy change only depends on the change in activation of one unit, and that the weight matrix is symmetric. Setting the diagonal weights to 0 corresponds to the assumption that biological neurons do not have self-connections. From a computational point of view, zeroing out the diagonal makes it more likely that the network will converge to one of the stored patterns rather than simply reproducing the input pattern.

Storage Capacity

In addition to knowing the circumstances under which a Hopfield network is guaranteed to converge, it is also useful to understand how many patterns may be stored in, and recalled from, such a network. Although more patterns may be stored if the pattern vectors are orthogonal, that structure cannot be assumed in general. Therefore, most results are based on the assumption that the patterns to be stored are random. Hopfield found experimentally that P, the number of binary patterns that can be stored and recalled with reasonable accuracy, is given (approximately) by $P = 0.15n$, where n is the number of neurons. For a similar DHNN, using bipolar patterns, it has been found (7) that $P = n/2 \log_2 n$.

As the number of stored patterns is increased, the network becomes less able to correctly recall the patterns; this is generally known as *network saturation.* Recall is also more difficult if the stored patterns are similar to each other (further from orthogonal).

CONTINUOUS HOPFIELD NETWORK

In contrast to the discrete form, the activations of the neurons in a continuous Hopfield network can take on a continuous range of values (most often between 0 and 1). The network dynamics are specified by differential equations for the change in activations. These differential equations are intimately connected to the underlying energy function for the network.

For a continuous Hopfield network, we denote the internal activity of a neuron as u_i; its output signal is $v_i = g(u_i)$, where g is a monotonically nondecreasing function of the input signal received by unit U_i. Most commonly g is taken to be the sigmoid function $v = 0.5[1 + \tanh(\alpha u)]$, which has range (0, 1). The parameter α controls the steepness of the sigmoid function. The differential equations governing the change in the internal activity of each unit are closely related to the energy function that will be minimized as the network activations evolve. Either the evolution equation, or the energy function, may be specified and the other relationship derived from it. A standard form for the energy function is

$$E = 0.5 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} v_i v_j + \sum_{i=1}^{n} \theta_i v_i \tag{3}$$

the corresponding evolution equation is:

$$\frac{d}{dt} u_i = -\frac{\partial E}{\partial v_i} = -\sum_{j=1}^{n} w_{ij} v_j - \theta_i \tag{4}$$

Thus the evolution, or relaxation, of a Hopfield network is based on gradient descent to minimize the energy function.

Continuous Hopfield neural networks that are used to solve constrained optimization problems have several standard characteristics. Each unit represents a hypothesis; the unit is "on" if the hypothesis is true and "off" if the hypothesis is false. The weights are fixed to represent both the constraints of the problem and function to be optimized. The solution of the problem corresponds to the minimum of the energy function. Each unit's activation evolves so that the energy function decreases.

In the next sections, we illustrate the use of CHNN for constraint satisfaction and constrained optimization, first for a very simple example, and then for the well-known N-queens and traveling salesman (TSP) problems.

Simple Example

To introduce the use of a CHNN, consider the network shown in Fig. 3, in which it is desired to have exactly one unit on. The weights must be chosen so that the network dynamics correspond to reducing the energy function.

In order to have a network that converges to a pattern of activations that solves a specified problem, it is common to design the energy function so that its minimum will be achieved for a pattern of activations that solves the given

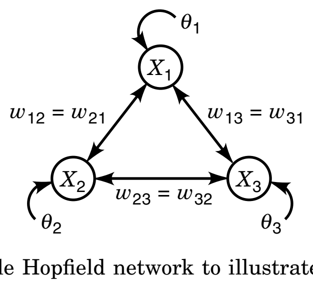

Figure 3. A simple Hopfield network to illustrate the interrelationship between the weights and the energy function.

problem. For this example, the energy function might be formulated as

$$E = C\left(1 - \sum_i v_i\right)^2$$

so that its minimum value (0) is achieved when exactly one of the units is on and the other two units each have activation of zero. Expanding the energy equation

$$E = C(1 - 2v_1 - 2v_2 - 2v_3 + v_1^2 + v_1v_2 + v_1v_3 + v_2v_1 \\ + v_2^2 + v_2v_3 + v_1v_3 + v_2v_3 + v_3^2)$$

and comparing it to the standard form given in Eq. (3) show that $\theta_i = -2C$ and $w_{ij} = C$. (Note that there is a self-connection on each unit; this does not interfere with the convergence analysis for a CHNN.) The energy function also includes a constant term C.

The differential equations governing the change in the internal activity u_i for each neuron are given by

$$\frac{d}{dt}u_i = -\frac{\partial E}{\partial v_i} = 2C[1 - (v_1 + v_2 + v_3)]$$

The *N*-Queens Problem

The problem of how to place eight queens on an 8-by-8 chessboard in mutually nonattacking positions was proposed in 1848 and has been widely studied since then. It is used as a benchmark for many methods of solving combinatorial optimization problems. In a neural network approach, one neuron is used for each square on the chessboard. The activation of the neuron indicates whether a queen is located on that square. Since a queen commands vertically, horizontally, and diagonally, only one queen should be present on any row or column of the board. The arrangement of the neurons for a smaller five-queens problem is shown in Fig. 4. Even for this small problem, the diagram would become very cluttered if we tried to show all of the connection pathways. To implement the energy function and evolution equations given in Eqs. 5 and 6, the units in each row and each column are fully interconnected; similarly the units along each diagonal and each antidiagonal are also fully interconnected (4).

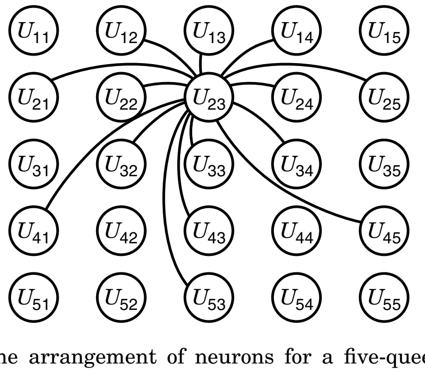

Figure 4. The arrangement of neurons for a five-queens problem. Connection pathways are shown only for unit U_{23}.

The constraints are

1. One and only one queen placed in each row
2. One and only one queen placed in each column
3. At most one queen placed on each diagonal

An energy function can be constructed for this problem, as follows:

$$\begin{aligned} E = {} & \frac{C_1}{2}\sum_x\sum_i\sum_{j\neq i} v_{xi}v_{xj} + \frac{C_2}{2}\sum_i\sum_x\sum_{y\neq x} v_{xi}v_{yi} \\ & + \frac{C_3}{2}\sum_x\left(\sum_i v_{xi} - 1\right)^2 + \frac{C_4}{2}\sum_i\left(\sum_x v_{xi} - 1\right)^2 \\ & + \frac{C_5}{2}\sum_x\sum_i \sum_{\substack{1\leq x+k;\, i+k\leq N \\ k\neq 0}} (v_{xi}v_{x+k;\, i+k}) \\ & + \frac{C_6}{2}\sum_x\sum_i \sum_{\substack{1\leq x+k;\, i-k\leq N \\ k\neq 0}} (v_{xi}v_{x+k;\, i-k}) \end{aligned} \tag{5}$$

The first constraint is represented by the first and third terms in the energy function; the second constraint is represented by the second and fourth terms in the energy function; the third constraint is represented by the fifth and sixth terms in the energy function (one term for the diagonal, and one for the antidiagonal). The corresponding motion equation for unit U_{xi} is

$$\begin{aligned} \frac{du_{xi}}{dt} = {} & -C_1\sum_{j\neq i} v_{xj} - C_2\sum_{y\neq x} v_{yi} - C_3\left(\sum_i v_{xi} - 1\right) \\ & - C_4\left(\sum_x v_{xi} - 1\right) - C_5\sum_x \sum_{\substack{1\neq x+k;\, i+k\leq N \\ k\neq 0}} v_{x+k;\, i+k} \\ & - C_6 \sum_{\substack{1\leq x+k;\, i-k\leq N \\ k\neq 0}} v_{x+k;\, i-k} \end{aligned} \tag{6}$$

One example of a valid solution to the five-queens problem is represented by the network configuration in which neurons U_{15}, U_{23}, U_{31}, U_{44}, and U_{52} are on and all others are off. For further discussion of CHNN solutions to this problem, see Refs. 4 and 8.

The Traveling Salesman Problem

The TSP is a well-known example of a class of computationally difficult problems for which the amount of time required to find an optimal solution increases exponentially as the problem size increases. In the TSP, every city in a given set of n cities is to be visited once and only once. A tour may begin with any city, and ends by returning to the initial city. The goal is to find a tour that has the shortest possible length.

With a Hopfield network, the TSP is represented by a $n \times n$ matrix of neurons in which the rows of the matrix represent cities and the columns represent the position in the tour when the city is visited. For example, if unit U_{24} is on for the TSP, it indicates that the second city is visited as the fourth stop on the tour. A valid solution is achieved when the network reaches a state of a permutation matrix, that is, exactly one

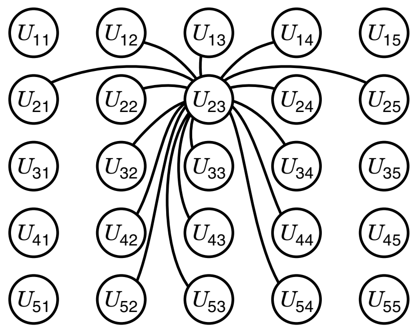

Figure 5. The arrangement of neurons for a five-city traveling salesman problem. Connection pathways are shown only for unit U_{23}.

unit on in each row and each column. The arrangement of the neuron for a five-city TSP is shown in Fig. 5, with connection pathways shown only for unit U_{23}. A widely used energy function for the TSP is

$$\begin{aligned} E = {} & \frac{C_1}{2}\sum_x\sum_i\sum_{j\neq i} v_{xi}v_{xj} + \frac{C_2}{2}\sum_i\sum_x\sum_{y\neq x} v_{xi}v_{yi} \\ & + \frac{C_3}{2}\sum_x\left(\sum_i v_{xi}-1\right)^2 + \frac{C_4}{2}\sum_i\left(\sum_x v_{xi}-1\right)^2 \\ & + \frac{C_5}{2}\sum_x\sum_{y\neq x}\sum_i d_{xy}v_{xi}(v_{y,i+1}+v_{y,i-1}) \end{aligned} \tag{7}$$

The first four terms in the energy function represent the validity constraints: the first term is minimized (zero) if each city is visited at most once. Similarly, the second term is zero if at most one city is visited at each stage in the tour. The third and fourth terms encourage each row and column in the network matrix to have one neuron on. The fifth term gives the value of the corresponding tour length. This term represents the TSP objective function. It is desired to make its value as small as possible while maintaining the validity of the tour.

To guarantee convergence of the network, the motion dynamics are obtained from the energy function according to the relationship

$$\begin{aligned} \frac{du_{xi}}{dt} = {} & -\frac{\partial E}{\partial v_{xi}} = -C_1\sum_{j\neq i} v_{xj} - C_2\sum_{y\neq x} v_{yi} - C_3\left(\sum_j v_{xj}-1\right) \\ & - C_4\left(\sum_y v_{yi}-1\right) - C_5\sum_{y\neq x} d_{xy}(v_{y,\,i+1}+v_{y,\,i-1}) \end{aligned} \tag{8}$$

where the internal activation u and the output signal v for any unit are related by the sigmoidal function $v = 0.5[1 + \tanh(\alpha u)]$.

For simulations, each neuron is updated using Euler's first-order difference equation

$$u_{xi}(t+\Delta t) = u_{xi}(t) + \left(\frac{du_{xi}}{dt}\right)\Delta t$$

The neurons' activations are initialized with random values, and the activations are allowed to evolve according to the governing equations for the network dynamics. The activations are updated iteratively until the network converges; the final configuration of activations gives the network's solution to the TSP.

The choice of network parameters has a significant effect on the quality of solutions obtained. The relative sizes of the coefficients in the energy equation influence the network either to emphasize valid tours (at the expense of tour length) or to seek short tours (which may not be valid). A very steep sigmoid function may force the network to converge quickly (but not necessarily to a good solution), while a shallow slope on the sigmoid function may result in the final activations not being close to 0 or 1.

Simulation Results. The energy function in the original presentation of a Hopfield network solution of the TSP was given as

$$\begin{aligned} E = {} & \frac{A}{2}\sum_x\sum_i\sum_{j\neq i} v_{xi}v_{xj} + \frac{B}{2}\sum_i\sum_x\sum_{y\neq x} v_{xi}v_{yi} \\ & + \frac{C}{2}\left(N - \sum_x\sum_i v_{xi}\right)^2 \\ & + \frac{D}{2}\sum_x\sum_{y\neq x}\sum_i d_{xy}v_{xi}(v_{y,i+1}+v_{y,i-1}) \end{aligned} \tag{9}$$

The third term in this form of the energy function encourages N neurons to be on, but does not try to influence their location. The original differential equation for the activity of unit U_{xi} was given by

$$\begin{aligned} \frac{d}{dt}u_{xi} = {} & -\frac{u_{xi}}{\tau} - A\sum_{j\neq i} v_{xj} - B\sum_{y\neq x} v_{yi} + C\left(N - \sum_x\sum_i v_{xi}\right) \\ & - D\sum_{y\neq x} d_{xy}(v_{y,\,i+1}+v_{y,\,i-1}) \end{aligned} \tag{10}$$

The first term on the right-hand side of this equation is a decay term, which can be motivated by analogy to electrical circuits but does not have a corresponding term in the energy equation. The parameter values that Hopfield and Tank used, namely,

$$A = B = 500, \qquad C = 200, \qquad D = 500,$$
$$N = 15, \qquad \alpha = 50, \qquad \tau = 1$$

give very little emphasis to the decay term, so the lack of corresponding energy term has relatively little significance. The parameter N must be taken to be larger than the actual number of cities in the problem to counterbalance the continuing inhibitory effect of the distance term; since the minimum of the distance component of the energy function is positive, the corresponding term in Eq. (10) acts to try to turn a unit off even when there are no constraint violations.

Although Hopfield and Tank (3) reported a very high rate of success in finding valid tours (16/20 trials) with about $\frac{1}{2}$ of the trials producing one of the two shortest tours, other researchers have been unable to match these results. The coordinates of the Hopfield and Tank 10-city test problem were generated randomly; the same locations have been used as a benchmark for other neural network solutions. Many variations have been investigated, including alternative energy functions, methods of choosing parameter values, and procedures for setting the initial activations.

Wilson and Pawley (9) provide a detailed statement of the Hopfield–Tank algorithm, together with an analysis of their experiments. Using the Hopfield–Tank parameters, with $\Delta t = 10^{-5}$, they found 15 valid tours in 100 attempts (45 froze and 40 failed to converge in 1000 epochs).

Wilson and Pawley tried a number of variations of the Hopfield and Tank algorithm, in attempting to obtain a success rate for valid tours which would approach that achieved by Hopfield and Tank. They experimented with different parameter values and different initial activity configurations and imposed a large distance penalty for visiting the same city twice, none of which helped much. Fixing the starting city helped on the Hopfield–Tank cities, but not on other randomly generated sets of cities.

One variation that did improve the ability of the network to generate valid tours was a modification of the initialization procedure. The Willshaw initialization is based on the rationale that cities on opposite sides of the square probably should be on opposite sides of the tour. The starting activity of each unit is biased to reflect this fact. Cities far from the center of the square received a stronger bias than those near the middle. The formula, in terms of the ith city and jth position, where the coordinates of the ith city are (x_i, y_i):

$$\text{bias}(i, j) = \cos\left[\arctan\left(\frac{y_i - 0.5}{x_i - 0.5}\right) + \frac{2\pi(j-1)}{n}\right] \sqrt{(x_i - 0.5)^2 + (y_i - 0.5)^2}$$

Although special analysis that relies on the geometry of the problem can improve the solution to the actual TSP, it does not generalize easily to other applications.

Issues

Proof of Convergence. For an energy function of the form of Eq. (3), the Hopfield network will converge if the activations change according to the differential equation given in Eq. (4), as the following simple calculations show. If $v_i = g(u_i)$ is monotonically nondecreasing, then $dv_i/du_i \geq 0$. Since

$$\frac{dE}{dt} = \sum_i \frac{dv_i}{dt}\frac{\partial E}{\partial v_i} = -\sum_i \frac{dv_i}{dt}\frac{du_i}{dt} = -\sum_i \frac{dv_i}{du_i}\frac{du_i}{dt}\frac{du_i}{dt}$$

the energy is nonincreasing, as required.

In the original presentation of the continuous Hopfield network (6) the energy function is

$$E = -0.5\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}v_iv_j - \sum_{i=1}^{n}\theta_i v_i + \frac{1}{\tau}\sum_{i=1}^{n}\int_0^{v_i} g_i^{-1}(v)\,dv$$

If the weight matrix is symmetric and the activity of each neuron changes with time according to the differential equation:

$$\frac{d}{dt}u_i = -\frac{u_i}{\tau} + \sum_{j=1}^{n} w_{ij}v_j + \theta_i \tag{11}$$

the network will converge. The argument is essentially the same as before.

Note that the weights must be symmetric for the equations given here to be valid. This symmetry follows from the fact that connections in a standard Hopfield network are bidirectional, that is, the connection from unit i to unit j and the connection from unit j to unit i are the same connection. Results for asymmetrical Hopfield networks are discussed below.

Choice of Coefficients. The relative importance assigned to each of the terms in the energy function plays a very important role in determining the quality of the solutions obtained. A variety of experimental investigations into the appropriate coefficients have been reported. Theoretical results have also been obtained; the choice of energy function coefficients is discussed further in the section on recent developments.

Local Minima. One shortcoming of the CHNN, as with any optimization procedure that always moves in the direction of improving the solution, is convergence to a local optima that is not the global optimum. One method of combating this difficulty is to add noise to the updating process, so that occasionally the network moves in a different direction. A general framework that includes the Boltzmann machine, Hopfield network, and others is known as the Gaussian machine (10). A Gaussian machine is described by three parameters: α (the steepness of the sigmoid function), T (temperature), and Δt (the time step). The operation of a Gaussian net consists of

Calculating the input to unit U_i:

$$\text{net}_i = \sum_{j=1}^{N} w_{ij}v_j + \theta_i + \epsilon$$

where ϵ is the random noise, which depends on temperature T.

Changing the activity level of unit U_i:

$$\frac{\Delta u_i}{\Delta t} = -\frac{u_i}{\tau} + \text{net}_i$$

Applying the output function:

$$v_i = f(u_i) = 0.5[1 + \tanh(\alpha u_i)]$$

The standard Hopfield neural network corresponds to $T = 0$ (no noise).

RECENT DEVELOPMENTS

Hopfield neural networks are being used for applications in many areas of engineering. The most recent examples can be found in conference proceedings, either for meetings that focus on neural network applications or for gatherings of researchers in a particular specialty. In the next sections, we consider some directions in which the basic Hopfield neural network model is being generalized. Methods of adapting the weights in HNN, both for CAM and optimization problems, are being developed. Investigation into HNN with nonsymmetric weights are giving theoretical results for conditions under which such network are guaranteed to converge. Research also continues into the determination of the storage capacity of the DHNN.

Adaptive Weights

Much of the neural network research has focused on networks in which either the activities of the neurons evolve or the strengths of the synapses (weights) adapt, but not both. However, a complete model of a biological process requires dynamical equations for both to specify the behavior of the system. On the other hand, applications of Hopfield networks to constrained optimization problems repeatedly illustrate the importance and difficulty of determining the proper weights to ensure convergence to a good solution. Progress is being made in both of these areas.

Learning Patterns. Dong (11) has developed an energy function for a system in which both activations and weights are adaptive and applied it to the study of the development of synaptic connection in the visual cortex. His dynamical equations for the activity of the neurons are essentially the same as given in Eq. (11). The adaptation of the weights follows a differential form of Hebbian learning, based on the "recent" correlation of the activities of the neurons that are on either end of the weighted pathway; this leads to Hebbian learning with a decay term. The weights remain symmetric throughout the process, so that the convergence analysis follows an energy function approach as described previously.

As a simple example, consider two neurons and the weight w on the connection path between them. Dong's dynamical equations for this illustrative special case are

$$\begin{aligned} a\frac{du_1}{dt} &= -u_1 + wv_1 \\ a\frac{du_2}{dt} &= -u_2 + wv_2 \\ v &= f(gu) \\ b\frac{ds}{dt} &= -s + v_1v_2 \\ w &= f(hs) \end{aligned}$$

The function f is piecewise linear, with a range between -1 and 1; that is, $f(x) = -1$ if $x \leq -1$, $f(x) = x$ if $-1 < x < 1$, $f(x) = 1$ if $x \geq 1$. The energy function is

$$E(v_1, v_2, w) = -wv_1v_2 + \frac{1}{2g}v_1^2 + \frac{1}{2g}v_2^2 + \frac{1}{2h}w^2$$

The origin (0,0,0) is a stable point, corresponding to unlearned connections and no neuron activity. If the constants g and h are greater than 1, the configurations (1,1,1), $(-1,-1,1)$, $(1,-1,-1)$, and $(-1,1,-1)$ are stable points. Each of these configurations has the property, which holds in general for stable points, that the weight on the connection is $\mathrm{sgn}(v_i v_j)$. The training of the network is conducted by presenting each pattern to be learned as the external input signal for a brief period of time, and cycling through the patterns until the weights have converged. The behavior of the system during learning depends on the strength of the external input to the system relative to the size of the weights between neurons. When the input signals dominate, the network can learn several input patterns; for weaker input signals, the network ultimately chooses only one of the patterns to memorize. These ideas provide the basis for a model of the first stage of cortical visual processing in mammals.

Constrained Optimization. The appropriate choice of the weights in a Hopfield net for constrained optimization has been the subject of much experimental work. It is well known that using larger values for the coefficients of the constraint terms helps guide the network toward valid solutions, but may result in poor quality solutions. On the other hand, increasing the value of the coefficient of the objective term helps to improve the quality of a solution, but it may result in an invalid solution because of a constraint violation.

Recently, Park (8) introduced a method for determining the coefficients of the energy function (and thereby the weights) adaptively as the network evolves. As the network evolves in the direction of minimization of the total energy, each term in the energy function competes with the other terms to influence the path to be followed. To find good coefficients for the energy function, the components of the energy are monitored and the coefficients adapted, depending on how far each component of the energy function is to its goal (minimum value), until a balanced relationship among the coefficients is reached. Using a steepest-*ascent* procedure with normalization, the coefficients are updated after every epoch of iteration until they reach a state of near equilibrium. While this may seem counterintuitive at first, it has the desired effect of increasing the coefficients of those terms that are contributing the most to the value of the energy function. It is those terms that most need to be reduced during network iteration. The final coefficient values are used to set the weight connections, and the network is run again to solve the problem.

A sample of the coefficient evolution for the 10-city TSP is illustrated in Fig. 6. In this example, the coefficient of the objective term (representing tour length) in the energy function, is fixed as $C_5 = 0.5$; the other coefficients (on the constraint terms) evolve subject to the restriction that $C_1 + C_2 + C_3 + C_4 = 1$. When the network was rerun with the converged coefficients, 94% of the trials resulted in valid tours; the length of the generated tours ranged from 2.69 to 3.84, with a mean length of 2.83. The efficacy of this method is even more striking on larger problems. Although the results vary depending on the choice of the fixed value for the coefficient of the objective term, 20- and 30-city problems

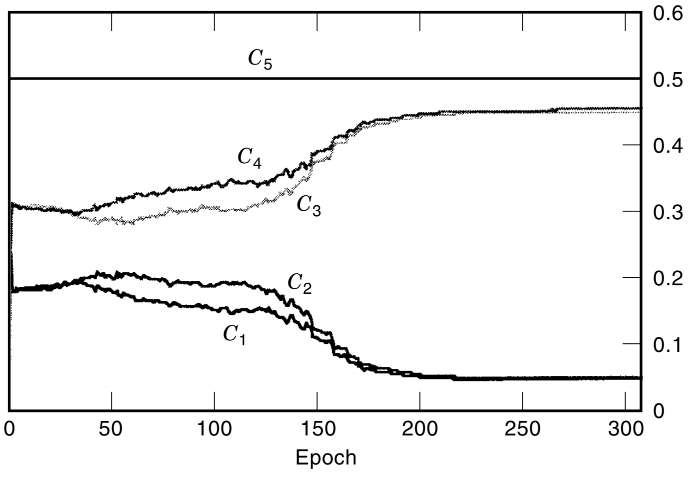

Figure 6. Evolution of coefficients on the constraint terms of the TSP; coefficient $C_5 = 0.5$.

(generated in a manner similar to that used by Hopfield and Tank for the 10-city problem) were successfully solved, with a high rate of valid solutions, for C_5 in the range of 0.2 to 0.5.

Storage Capacity

Another area of active research for Hopfield networks used as CAM is the storage capacity of the network. Many investigations are based on the assumption that the patterns are random (independent, identically distributed uniform random variables.) The question is, how many patterns (vectors with components of $+1$ or -1) can be stored and retrieved (from a minor degradation of a stored pattern)? A small fraction of errors may be allowed in the retrieved pattern. Hopfield suggested, based on numerical simulations, that P, the number of patterns that can be stored and retrieved, is given by $P = cn$ with $c = 0.15$.

More recently, Martingale techniques have been applied (12) to a different joint distribution of the spins (patterns), extending the theoretical results to situations beyond those investigated previously (7). Assuming that the patterns have the same probability distribution, are orthogonal in expectation, and are independent, Francois shows that there are energy barriers (which depend on d, the acceptable fraction of errors in the reconstructed pattern) surrounding each memorized pattern. For almost perfect recall ($d = 1/n$) the storage capacity can be as large as $c = [2(1 + g) \ln n]^{-1}$ with $g > 2$.

Stability Results

Investigations into the stability of more general Hopfield-type models have considered asynchronous updates for a continuous-valued, discrete-time Hopfield model (13) and the design of a continuous-valued, continuous-time analog-to-digital converter (14). In general, stability arguments rely on sophisticated mathematical theory and are not easily summarized in a brief presentation.

One approach to the investigation of asynchronous updates is based on the Takeda-Goodman synchronous model:

$$x(k+1) = TF(x(k)) + (I - B)x(k) + u$$

where T is the interconnection matrix of the neural network (usually assumed symmetric), F is a diagonal nonlinear function (usually assumed monotonic, often sigmoidal), and u is a vector of inputs (assumed constant). With the further assumptions that T is D-stable (see definition in the next paragraph), F is continuously differentiable, and $B = I$, the previous stability results have been extended by considering a class of desynchronizations which satisfy a mild regularity assumption (this is known as "absolute stability" in the Russian literature) (13).

Another example of recent extensions to the basic Hopfield model is the design of a continuous-valued, continuous-time analog to digital converter, based on a strictly lower triangular interconnection matrix. This structure leads to a unique globally stable equilibrium that allows the network to function as an error-free analog-to-digital converter (since there are no spurious stable states). The stability analysis is not based on an energy function, but uses instead the notion of D-stability. The matrix A is called D-stable if A is an $n \times n$ real asymptotically stable matrix and the product AD is asymptotically stable for any diagonal matrix D that has diagonal elements in the interval $[-1,1]$ (14).

Asymmetric Weights

The stability of asymmetric Hopfield networks is of practical interest, both for more general models (e.g., connectionist expert systems) and for the implementation of theoretically symmetric networks (since it is almost impossible to preserve the symmetry of the connections exactly in hardware).

Many results for nonsymmetric connections depend on the absolute value of the weights; however, these may be overly restrictive. For example, if $w_{ij} = -w_{ji}$ for all i,j, the network is absolutely stable, but results relying on absolute value considerations will not establish the fact. It has also been shown that if the largest eigenvalue of $W + W^T$ is less than 2, then the network is absolutely stable. A more convenient corollary of this result is that if

$$\sum_{i;\,j} (w_{ij} + w_{ji})^2 < 4$$

then the network is absolutely stable (15).

To study computational models based on asymmetric Hopfield-type networks, a classification theory for the energy functions associated with Hopfield networks has been introduced, and convergence conditions deduced for several different forms of asymmetric networks. For example, two networks have been developed, using a triangular structure, to solve the maximum independent set of a graph problem. Although this problem can be solved with a standard Hopfield network, the triangular network is a more simple and efficient procedure. See Ref. 16 for details.

SUMMARY AND CONCLUSIONS

Hopfield neural networks comprise a rich and varied realm of the overall field of artificial neural networks. Applications can be found in many areas of engineering. Continuing investigation into the theoretical and practical considerations governing the convergence properties of the networks provide a firm foundation for the use of Hopfield models and their extension to more generalized settings. Work continues on differences in performance that may occur when the networks are implemented with fully parallel (asynchronous) updating of the activations.

BIBLIOGRAPHY

1. J. A. Anderson and E. Rosenfeld, *Neurocomputing: Foundations of Research,* Cambridge, MA: MIT Press, 1988.
2. D. W. Tank and J. J. Hopfield, Collective computation in neuronlike circuits, *Sci. Am.,* **257** (6): 104–114, 1987.
3. J. J. Hopfield and D. W. Tank, Computing with neural circuits: a model, *Science,* **233** (8): 625–633, 1986.
4. Y. Takefuji, *Neural Network Parallel Computing,* Boston: Kluwer, 1992.
5. J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, *Proc. Natl. Acad. Sci.,* **79**: 2554–2558, 1982.

6. J. J. Hopfield, Neurons with graded response have collective computational properties like those of two-state neurons, *Proc. Natl. Acad. Sci.,* **81**: 3088–3092, 1984.
7. R. J. McEliece et al., The capacity of the Hopfield associative memory, *IEEE Trans. Inf. Theory,* **IT-33**: 461–482, 1987.
8. C-Y. Park, Energy landscape analysis of the performance of Hopfield neural networks as a method of solving combinatorial optimization problems, Ph.D. dissertation, Florida Institute Technol., Melbourne, FL, 1996.
9. G. V. Wilson and G. S. Pawley, On the stability of the traveling salesman problem algorithm of Hopfield and Tank, *Biol. Cybern.,* **58**: 63–70, 1988.
10. Y. Akiyama et al., Combinatorial optimization with Gaussian machines, *Intl. Joint Conf. Neural Netw.,* Washington, DC, 1989, vol. I, pp. 533–540.
11. D. Dong, Dynamic properties of neural networks, Ph.D. dissertation, California Institute of Technol., Pasadena, CA, 1991.
12. O. Francois, New rigorous results for the Hopfield's neural network model, *Neural Netw.,* **9**: 503–507, 1996.
13. A. Bhaya, E. Kaszkurewics, and V. S. Kozyakin, Existence and stability of a unique equilibrium in continuous-valued discrete-time asynchronous Hopfield neural networks, *IEEE Trans. Neural Netw.,* **7**: 620–628, 1996.
14. G. Avitable et al., On a class of nonsymmetrical neural networks with application to ADC, *IEEE Trans Circuits Syst.,* **CAS-38**: 202–209, 1991.
15. K. Matsuoka, Stability conditions for nonlinear continuous neural networks with asymmetric connection weights, *Neural Netw.,* **5**: 495–500, 1992.
16. Z-B. Xu, G-Q. Hu, and C-P. Kwong, Asymmetric Hopfield-type networks: theory and applications, *Neural Netw.,* **9**: 483–501, 1996.

LAURENE V. FAUSETT
University of South Carolina—Aiken

NEURAL NETWORK CHIPS. See NEURAL CHIPS.

NEURAL NETWORKS. See FUNCTION APPROXIMATION; NEURAL NET APPLICATIONS; NONCONVENTIONAL COMPUTERS; OPTICAL NEURAL NETS.

NEURAL NETWORKS FOR FEEDBACK CONTROL. See NEURAL NETS FOR FEEDBACK CONTROL.