On the Dynamics of Discrete-Time, Continuous-State Hopfield Neural Networks

Lipo Wang

Abstract—We propose answers to four open problems listed recently by Koiran on the dynamics of discrete-time, continuous-state Hopfield neural networks.

Index Terms—Convergence, cycles, fixed point, Hopfield neural networks, oscillation, stability.

I. INTRODUCTION

In his famous paper [1], Hopfield proposed an energy function for the discrete-time, discrete-state Hopfield neural network (HNN), and showed that this energy function decreases for any change of neuronal states. Hopfield further showed that a different energy function decreases if any neuron changes its state in the continuous-time, continuous-state HNN [2]. Fogelman-Soulie et al. [3] and Marcus and Westervelt [4] proved that the latter energy function also decreases for any neuronal state changes in an HNN if neuronal states are continuous; however, the dynamics is of discrete time. It might have largely been taken for granted that, because of the existence of such an energy function, the discrete-time, continuous-state HNN should approach a stable state (fixed point) or a length-2 cycle (periodic oscillation between two states), until Koiran [5] published a number of results recently.

Koiran [5] first proved that the discrete-time, continuous-state HNN does approach a fixed point if the network has a finite number of fixed points and the energy function is bounded from below. Koiran [5] then showed that almost every discrete-time, continuous-state HNN has a finite number of fixed points if the neurons are updated asynchronously (serial mode of operation). Koiran [5] listed the following open problems for the discrete-time, continuous-state HNN, simifying the need to improve our understanding of this type of HNN’s. In this paper, we attempt to answer these open problems (with minor changes).

1) Prove that the number of length-2 cycles is finite for almost every network if the neurons are updated synchronously (parallel mode of operation).

2) Give a condition on the neuronal response function f, ensuring that every network has a finite number of fixed points in serial mode and a finite number of length-2 cycles in parallel mode of operation.

3) Give an upper bound on the number of fixed points or length-2 cycles when it is finite.

4) Does the network still converge if it has an infinite number of fixed points or length-2 cycles?

An intuitive reason that the dynamics of the discrete-time, continuous-state HNN is nontrivial may be stated as follows. To prove that the discrete-time, discrete-state HNN must stabilize itself after a finite number of iterations starting from any initial condition, we need the following two conditions, in addition to the existence of a decreasing energy function: 1) the energy function is bounded from below [1], and 2) each time the energy function decreases, it must decrease by at least some minimum amount [2]. Both conditions are easily derived from the finite size of the discrete-time, discrete-state HNN [6]; however, condition 2) no longer holds if the state of a neuron is continuous.

II. ANSWERS TO THE FOUR OPEN PROBLEMS

In the discrete-time, continuous-state HNN, the output \( V_i \) of neuron \( i \) is determined by its input \( U_i \) (Fig. 1):

\[
V_i(t + \Delta t) = f[U_i(t) - \theta_i]
\]

where \( \theta_i \) is the firing threshold of neuron \( i \), \( i = 1, 2, \ldots, n \). \( f \) is a continuous increasing function so that \( f^{-1} \) exists, and the inputs to the neurons are

\[
U(t) = WV(t)
\]

with \( U^T = (U_1, U_2, \ldots, U_n) \), \( V^T = (V_1, V_2, \ldots, V_n) \), and a symmetric weight matrix \( W = W^T \) with nonnegative diagonals \( W_{ii} \geq 0 \), \( i = 1, 2, \ldots, n \). The energy function of the network is [2]-[4]

\[
E = -\frac{1}{2} V^T W V + \theta^T V + \sum_i \int_0^{U_i} f^{-1}(\xi) \, d\xi.
\]

The first problem is readily solved by combining the corresponding result for the serial mode of operation stated above [5] and the following result proven by Bruck and Goodman [7]; for any parallel mode of operation in a neural network described by \( N = (W, \theta) \), an equivalent serial mode of operation can be found in another network.
formed as follows: \( \bar{N} = (\bar{W}, \bar{\theta}) \), where
\[
\bar{W} = \begin{pmatrix} 0 & W \\ W & 0 \end{pmatrix} \quad \text{and} \quad \bar{\theta} = \begin{pmatrix} \theta \\ \theta \end{pmatrix}
\]

Any length-2 cycle in network \( \bar{N} \) is, in fact, equivalent to a fixed point in network \( \bar{N} \). From now on, we will implicitly consider the serial mode of operation only. The results for the parallel mode of operation will follow due to the above relationship between the two modes of operation.

We now discuss the second problem. The fixed points of the system are the solutions of the following equations:
\[
W_{11} V_1 + W_{12} V_2 + \cdots + W_{1n} V_n = f^{-1}(V_1) + \theta_1 \\
W_{21} V_1 + W_{22} V_2 + \cdots + W_{2n} V_n = f^{-1}(V_2) + \theta_2 \\
\vdots \\
W_{n1} V_1 + W_{n2} V_2 + \cdots + W_{nn} V_n = f^{-1}(V_n) + \theta_n .
\]
(4.1)–(4.n)

Since the left-hand sides of (4.1)–(4.n) are linear in \( V \) and represent planes in the \( n \)-dimensional space spanned by \( V \), it is possible to find an infinite number of solutions for these equations, the right-hand sides of (4.1)–(4.n) must intersect with these linear planes at an infinite number of points, i.e., there exists a set of constants \( \{c_1, c_2, \cdots, c_n\} \) such that \( f^{-1}(V_i) + \theta_i = c_i V_i \) holds for an infinite number of points \( V_i \in [-1, 1] \) (Fig. 1), for all \( i = 1, 2, \cdots, n \). This, in turn, is possible only if \( \theta_1 = \theta_2 = \cdots = \theta_n \equiv \theta \), that is, all neurons have the same firing threshold, and thus \( c_1 = c_2 = \cdots = c_n \equiv c \).

In this case, (4.1)–(4.n) become a set of linear equations. Hence, an infinite number of fixed points exists if and only if there exists a constant \( c \) such that the following three conditions are simultaneously satisfied: 1) all neurons have the same firing threshold, i.e., \( \theta_1 = \theta_2 = \cdots = \theta_n \equiv \theta \); 2) \( f^{-1}(V_i) + \theta = c V_i \) holds for an infinite number of points \( V_i \in [-1, 1] \) (Fig. 1); and 3) the rank of matrix \( M \equiv (W - cI) \) is less than \( n \), where \( I \) is the \( n \)-th order unit matrix.

The example given by Koiran [5] is a special case with \( \theta_1 = \theta_2 = \cdots = \theta_n \equiv \theta \equiv 0, c = 1 \), and \( f^{-1}(V_i) = \theta \), for all \( V_i \in [-1, 1] \), \( f(V_i) = 1 \) for \( V_i > 1 \), \( f(V_i) = -1 \) for \( V_i < -1 \); however, no conditions for synaptic weights \( W \) were specified in [5]. We now show that not every \( W \) in Koiran’s example satisfies condition 3) above or leads to an infinite number of fixed points. Consider an HNN of two neurons, i.e., \( n = 2 \), \( W_{12} = W_{21} = 3 \), and \( W_{11} = W_{22} \equiv \theta_1 = \theta_2 = 0 \). If the network state starts initially from any \( 0 < V_1 < 1 \) and \( 0 < V_2 < 1 \), it is straightforward to verify that the network will settle down at \( V_1 = V_2 = 0 \). Similarly, if the network starts initially from any \(-1 < V_1 < 0 \) and \(-1 < V_2 < 0 \), it will stabilize at \( V_1 = V_2 = -1 \). If the initial states of the network are such that \( V_1 = 0 \) and \( 0 < V_2 < 1 \), the final states of the network will be: 1) \( V_1 = V_2 = 1 \) if neuron 1 is updated first, or 2) \( V_1 = V_2 = 0 \) if neuron 2 is updated first. It is now clear that the only possible fixed points for this network are \( V_1 = V_2 = 1 \), \( V_1 = V_2 = -1 \) and \( V_1 = V_2 = 0 \); the number of fixed points in this network is finite.

Let us study a slightly different two-neuron network. Suppose \( W_{11} = W_{22} = \theta_1 = \theta_2 = 0 \), \( W_{12} = W_{21} = 1 \), and the input–output response function of the neurons is such that (Fig. 1) \( f(V_i) = V_i \) at \( V_i = 1/m \), with \( m = \pm 1, \pm 2, \cdots, \pm \infty \), and \( i = 1, 2 \). One can verify that if we select \( c = 1 \) again, all conditions 1)–3) given above are satisfied, and that \( V_1 = V_2 = 0 \) and \( 1/m \), with \( m = \pm 1, \pm 2, \cdots, \pm \infty \), are the fixed points of the network. This example is interesting since an infinite number of fixed points exists, and yet all but one fixed point \( (V_1 = V_2 = 0) \) is isolated.

A variation of this example, with \( m \) being finite, in fact leads to an answer to the third problem: there does not exist an upper bound for the number of fixed points or length-2 cycles when it is finite.

Now, let us consider the fourth problem. In general, in a dynamic system with a bounded Lyapunov function which strictly decreases along all trajectories, nor all trajectories necessarily converge to fixed points if there exists an infinite number of fixed points. The following example may be used to illustrate this point. In polar coordinates, consider
\[
g(r, \alpha) = \tanh \left( (1-r^2)^2 \right)
\]
(5)

Function \( g \) is minimum at \( r = 1 \) (the unit circle), and is bounded for all \( \alpha \) and \( r \geq 0 \). The \textit{spiral flow} defined by
\[
\frac{dr}{dt} = -\frac{\partial g}{\partial r} \quad \text{and} \quad \frac{d\alpha}{dt} = 1
\]
approaches the unit circle asymptotically, i.e.,
\[
|1 - r| \approx e^{-t/4}, \quad \text{as } t \to +\infty \text{ and } r \to 1
\]
(7)

and monotonously decreases \( g \); however, the flow spirals around the unit circle indefinitely, and does not approach any point on the unit circle! A discrete-time (iterative) version of this example can be obtained by the standard \textit{transition map} of the above continuous flow [8]. A \textit{gradient flow} defined by
\[
\frac{dr}{dt} = -\frac{\partial g}{\partial r} \quad \text{and} \quad \frac{d\alpha}{dt} = -\frac{\partial g}{\partial \alpha} = 0
\]
(8)

where no flow components tangent to the gradient exist, does lead to fixed points on the unit circle. The dynamics of a neural network described by (1) and (2) happens to be such a \textit{gradient flow}, and hence it must converge to a fixed point regardless of whether or not the number of fixed points is infinite: a combination of (1)–(3) gives
\[
\Delta f^{-1}(V_i) \equiv f^{-1}[V_i(t + \Delta t)] - f^{-1}(V_i(t)) = -\frac{\partial E}{\partial V_i}, \quad \text{for all } i.
\]
(9)

III. CONCLUSIONS

In summary, we have shown that the number of length-2 cycles is finite for \textit{almost} every network if the neurons are updated \textit{synchronously} (parallel mode of operation). We have given a condition on the neuronal response function \( f \), ensuring that every network has a finite number of fixed points in the serial mode or length-2 cycles in the parallel mode of operation: there does not exist a constant \( c \) such that the following three conditions are simultaneously satisfied: 1) all neurons have the same firing threshold, i.e., \( \theta_1 = \theta_2 = \cdots = \theta_n \equiv \theta \); 2) \( f^{-1}(V_i) + \theta = c V_i \) holds for an infinite number of points \( V_i \in [-1, 1] \); and 3) the rank of matrix \( M \equiv (W - cI) \) is less than \( n \), where \( I \) is the \( n \)-th order unit matrix. We have demonstrated
that there does not exist an upper bound on the number of fixed points or the number of length-2 cycles when this number is finite. We have shown that the network converges to a fixed point (serial mode) or a length-2 cycle (parallel mode) regardless of whether or not the network has an infinite number of fixed points or length-2 cycles.

ACKNOWLEDGMENT

The author thanks A. F. Ivanov and D. L. Elliott for many stimulating discussions, and for their help in constructing the example shown in (5) and (6). Many helpful comments and suggestions from the reviewers are also gratefully appreciated.

REFERENCES


A Fast Training Algorithm for Neural Networks

Jarosław Bilski and Leszek Rutkowski

Abstract—The recursive least squares method (RLS) is derived for the learning of multilayer feedforward neural networks. Simulation results on the XOR, 4-2-4 encoder, and function approximation problems indicate a fast learning process in comparison to the classical and momentum backpropagation (BP) algorithms.

Index Terms—Learning systems, least squares methods, multilayer perceptrons, neural networks, recursive estimation.

I. TERMINOLOGY

In the subsequent sections of this brief, the following terminology will be used.

\[ L \]
Number of layers in the network.

\[ N_k \]
Number of neurons in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ N_0 \]
Inputs to the network.

\[ \gamma, \delta \]
Constants.

\[ \mu \]
Learning coefficient in the backpropagation (BP) algorithm.

\[ \alpha \]
Vector of input signals of the neural network.

\[ y_i^{(k)} \]
Output signal of the \( i \)th neuron, \( i = 1, \ldots, N_k \); in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ x_i^{(k)} \]
Input, \( i = 0, \ldots, N_{k-1} \) for \( k \)th layer, \( k = 1, \ldots, L \).

\[ w_i^{(k)} \]
Vector of weights of the \( i \)th neuron, \( i = 1, \ldots, N_k \) in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ s_i^{(k)} \]
Vector of inputs in the layer, \( k = 1, \ldots, L \).

\[ d_i^{(k)} \]
Desired output of the \( i \)th neuron, \( i = 1, \ldots, N_k \) in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ e_i^{(k)} \]
Error of the \( i \)th neuron, \( i = 1, \ldots, N_k \) in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ f_i^{(k)} \]
Desired linear summation output of the \( i \)th neuron, \( i = 1, \ldots, N_k \) in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ g_i^{(k)} \]
Vector of the error signals in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ [u_1, \ldots, u_{N_0}]^T \]
Vector of input signals of the neural network.

\[ [y_1^{(k)}, \ldots, y_{N_k}^{(k)}] \]
Output signals of the neural network in the \( k \)th layer.

\[ s_0^{(k)} \]
Summation output of the \( k \)th layer.

\[ [s_1^{(k)}, \ldots, s_{N_k}^{(k)}] \]
Vector of summation outputs in the \( k \)th layer.

\[ f^{-1}(y_i^{(k)}) \]
Error of the \( i \)th neuron, \( i = 1, \ldots, N_k \) in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ f_i^{(k)} \]
Desired linear summation output of the \( i \)th neuron, \( i = 1, \ldots, N_k \) in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ e_i^{(k)} \]
Error of the \( i \)th neuron, \( i = 1, \ldots, N_k \) in the \( k \)th layer, \( k = 1, \ldots, L \).

\[ [x_0^{(k)}, \ldots, x_{N_k}^{(k)}] \]
Input signals of the \( k \)th layer, \( k = 1, \ldots, L \).

In Fig. 1, we show a model of the \( i \)th neuron in the \( k \)th layer.

II. INTRODUCTION

The BP [6] is the most widely applied multilayer neural-network learning algorithm. Unfortunately, it suffers from a number of shortcomings. One such shortcoming is the slow convergence. Therefore, several approaches have been developed [4], [5], [8] in order to speed up the convergence.

The method presented in this brief relies on the analogy between adaptive filters and neural networks. In adaptive filtering, the RLS algorithm is typically an order of magnitude faster than the LMS.