Evolution of Random Synaptic Weights of the Hopfield Associative Memory: How Chaotic Trajectories Turn into Fixed Point Attractors?

Akira Imada[1] **and Keijiro Araki**[2]

[1]Graduate School of Information Science, Nara Institute of Science and Technology, Takayama, Ikoma, Nara, Japan
E-mail: akira-i@is.aist-nara.ac.jp

[2]Graduate School of Information Science, and Electrical Engineering, Kyusyu University, Kasuga, Fukuoka, Japan
E-mail: araki@c.csce.kyusyu-u.ac.jp

Abstract

We apply evolutionary computations to Hopfield's neural network model of associative memory. We reported elsewhere that a fully connected neural network with random synaptic weights evolves to create fixed point attractors exactly at the locations of patterns to be memorized by a genetic algorithm. In this paper, we present the process of the evolution from chaotic behaviors to an associative memory.

1 Introduction

Associative memory is a dynamical system which has a number of stable states with a domain of attraction around them [Komlós and Paturi, 1988]. If the system starts at any state in the domain, it will converge to the stable state. Hopfield [1982] proposed a fully connected neural network model of associative memory in which we can store information by distributing it among neurons and recall it from its noisy and/or partial input.

The dynamical behavior of its neuron states strongly depends on the values of synaptic strength (weight) between neurons. Hopfield used the Hebbian rule [Hebb, 1949] to prescribe these weight values. We explore it with a genetic algorithm (GA) [Holland, 1975, Goldberg, 1989].

Previously, we reported that a fully connected neural network with random synaptic weights evolves to create fixed point attractors exactly at the locations of patterns to be memorized by a GA [Imada and Araki, 1995, Imada and Araki, 1997]. Here, we study the evolution in more detail by observing trajectories of network state as snapshots.

2 The Hopfield Model

The model consists of N neurons and N^2 synapses where each neuron can be in one of two states ± 1. The network of these neurons memorized p bipolar patterns:

$$\xi^\nu = (\xi_1^\nu, \cdots, \xi_N^\nu), \qquad \nu = 1, \cdots, p$$

as equilibrium states, where ξ_i^ν takes the value of either 1 or -1. Hopfield employed a discrete-time, asynchronous update scheme. Namely, at most one neuron updates its state at a time[1], according to

$$s_i(t+1) = sgn\left(\sum_{j \neq i}^{N} w_{ij} s_j(t)\right),$$

where $s_i(t)$ is a state of i-th neuron at time t. The behaviors of the collective states of individual neurons are characterized by the synaptic weights. When these synaptic weights are determined appropriately, the network stores some number of patterns as fixed points. Hopfield specified these w_{ij}'s by the Hebbian rule [Hebb, 1949], i.e.,

$$w_{ij} = \frac{1}{N}\sum_{\nu=1}^{p} \xi_i^\nu \xi_j^\nu \quad (i \neq j), \quad w_{ii} = 0.$$

Then giving one of the memorized patterns to the network as an initial state, possibly including a few errors, will result in the stable state after certain time steps of updating.

3 GA Implementation

In this simulation, a weight matrix R_{ij} is produced randomly before the start of the GA run, and remains unchanged during the evolution.

In each generation, chromosomes, made up of mostly 1 but a few of -1 and 0 (allele), modify this initial matrix R_{ij} by multiplying the alleles to the component of the matrix and produce a population of weight matrices. Then the

[1]Though Hopfield chose one neuron at a time *randomly*, in this paper, a neuron is chosen according to pre-assigned order (once in a cycle).

obtained matrices are evaluated their capability of memorizing the given patterns (fitness value). According to the fitness values, two parent chromosomes are selected to be recombined to make one offspring. The offspring is mutated occasionally and reconstructs the next generation.

The cycle of reconstructing the new population with better individuals and restarting the search is repeated until a perfect solution is found or a maximum number of generation has been reached. The specific details are as follows.

(1) (initialization) Chromosomes are N^2-dimensional vectors, and they are initialized so that their components are randomly selected from $\{1, 0, -1\}$ with the probability of .98, .01 and .01, respectively.

(2) (fitness evaluation) When ξ^ν, one of the patterns to be stored, is given to the network as an initial state, the state of neurons varies from time to time afterwards (unless ξ^ν is a fixed point). In order for the network to function as an associative memory, the instantaneous state of the neurons must be similar to the initial state. The similarity as a function of time is defined by,

$$m^\nu(t) = \frac{1}{N} \sum_{i=1}^{N} \xi_i^\nu s_i^\nu(t),$$

where $s_i^\nu(t)$ is the state of the i-th neuron at time t. This is referred to as *overlap* by convention. In evaluating the fitness value, the temporal average of the overlap $\langle m^\nu \rangle$ is calculated for each stored pattern. Then they are also averaged over all patterns. That is, the fitness value f is

$$f = \frac{1}{t_0 \cdot p} \sum_{t=1}^{t_0} \sum_{\nu=1}^{p} m^\nu(t).$$

In this paper, t_0 is set to $2N$, twice the number of neurons. Note that the fitness 1 implies all the p patterns are stored as fixed points, while fitness less than 1 includes many possible other cases.

(3) (selection) Two parent chromosomes are chosen *randomly* from the best 40% of the population to be recombined.

(4) (recombination) Recombinations are made with *uniform crossover* [Syswerda, 1989], operating on alleles of the selected parents' chromosomes, i.e., two parents $(u_1, \cdots, u_n)$ and $(v_1, \cdots, v_n)$ produce an offspring $(w_1, \cdots, w_n)$ such that w_i is either u_i or v_i with equal probability.

(5) (mutation) Mutation is made by rotating the allele with the probability 0.01 as follows.

$$\{1\} \rightarrow \{-1\}, \quad \{-1\} \rightarrow \{0\}, \quad \{0\} \rightarrow \{1\}$$

The procedures (3)–(5) are repeated until all the individuals in the worst 60% of the population are replaced with the offspring.

4 Experimental Results

Evolution of Random Synaptic Weights

Since the initial matrix is completely random, all the networks in the first generation do not function as an associative memory. The goal of the GA is to find the optimal combination of alleles $c_{ij} \in \{1, 0, -1\}$ in chromosome which modifies the initial matrix R_{ij} by multiplying the c_{ij} to the corresponding component of the matrix.

All the simulations in this paper were carried out on networks with 49 neurons.

First, the effect on evolution of varying p, the number of given patterns, is studied. We repeat each simulation 30 times with different random number seed for the same p. If we find the perfect solution(s) then we increment p. Thus, we found a matrix evolved to store a maximum of seven patterns. In Figure 1, we show the representative sample of the best fitness versus generation for these seven patterns. In this case, at the 2312-th generation a network emerges that stores all the seven patterns as fixed points.

In the next two subsections, we present the dynamical behavior of this evolution.

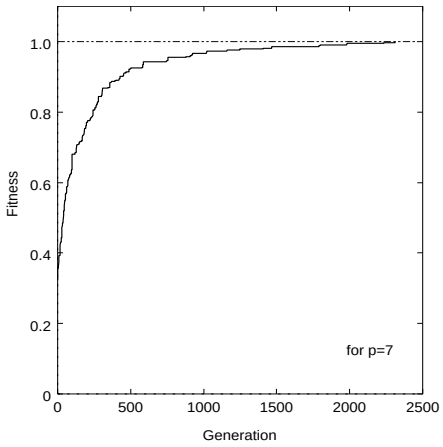

Figure 1: **Fitness of the Best of Generation**

How the Chaotic Trajectories Turn to Attractors?

In the Hopfield model of associative memory, an initial state of the network determined by an input is to be updated from time to time. We can observe this by the temporal expansion of the Hamming distance between an instantaneous network state and the input.

In Figure 2, the Hamming distances of instantaneous network state from the input are plotted against updating time as snapshots during the evolution.

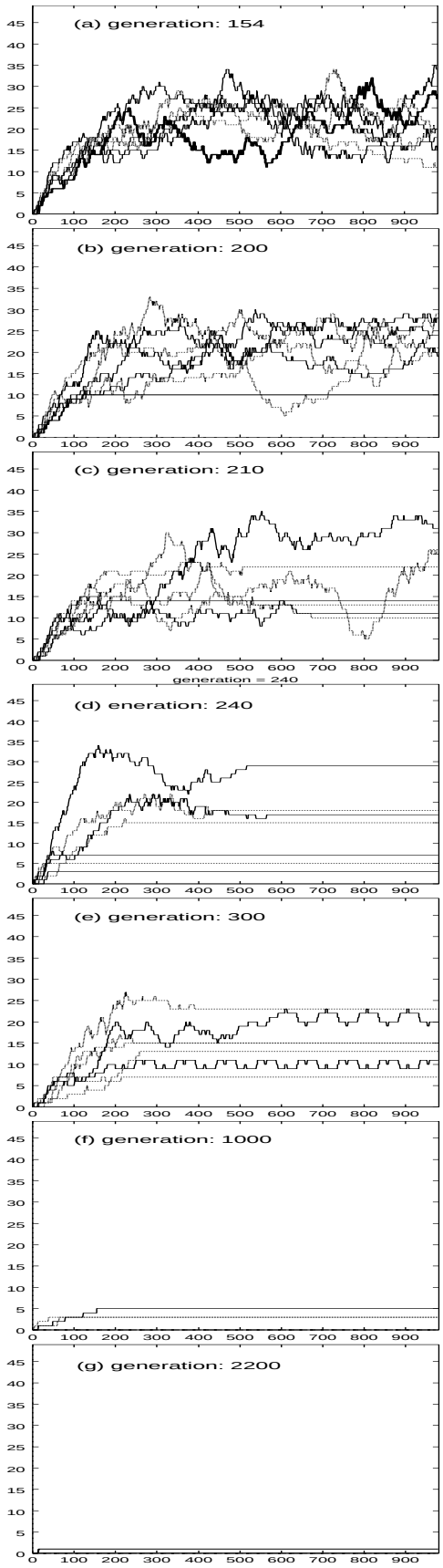

Figure 2: **Trajectories Resulted from 7 Inputs.**

Since we started with random synaptic weights, any input of the given patterns results in the chaotic trajectory in an early generation (Figure 2-a). Then, an attractor appears (Figure 2-b). Thereafter, the number of the attractors increases (Figure 2-c). As evolution proceeds further, chaotic trajectories disappear (Figure 2-d). In the mean time, some limit cycle trajectories occasionally appear and disappear (Figure 2-e). In the last stage of the evolution, the location of the most of these attractors are shifted towards the patterns to be memorized. At generation 1000, for example, the locations of five out of seven attractors coincide with the given patterns, and the other two locations are within 5 Hamming distances from the patterns (Figure 2-f). It requires a long time for these near-fixed-point-attractors to disappear (Figure 2-g). Finally, at generation 2312, seven fixed point attractors are created exactly at the location of each given pattern.

There Still Remains Spurious Attractors

We observed the basin size of the network as follows. An input is randomly sampled from the given patterns. After d bits are flipped at random, this is given to the network. The temporal average of the *overlaps* between the input and the network state visited by the dynamics is calculated. In Figure 3, the averaged overlap over 800 runs are plotted against noisy-bit d. We also show the result of the Hebb rule associative memory which stores the same seven patterns.

As shown in the figure, the obtained matrix has comparatively small basin of attraction. This is probably due to remaining spurious attractors around the created memories.

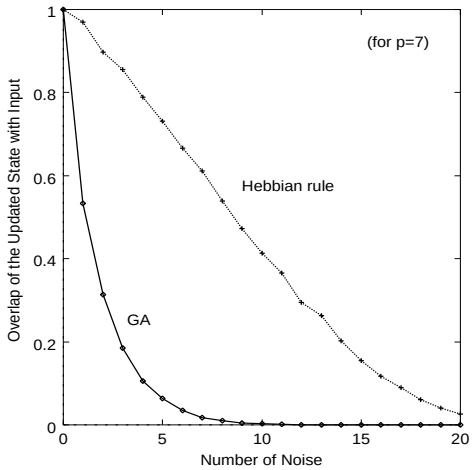

Figure 3: **Basin of Attraction**

5 Conclusions

We have shown that a fully connected neural network with random synaptic weights evolves to function as an associative memory. This is by *adaptively* pruning some of the connections and reversing some excitatory/inhibitory connections. Before the GA runs, the network shows chaotic behavior because of its random synaptic weights. As evolution proceeds, these chaotic trajectories turn to attractors, though their locations are apart from the patterns to be memorized. Then, they gradually approach to the location of the given patterns. Finally, all locations of these attractors coincide with the given patterns. We have presented this process by showing input trajectories during the evolution.

It is surprising to see that the learning of the patterns is only by finding the optimum combination of 1, 0, -1 to be multiplied to the component of a randomly chosen weight matrix R_{ij}. The absolute values of the synaptic weights remain unchanged except that some of them are zeroed. In [Imada and Araki, 1997], we conjectured that the evolution is partly by destabilizing spin-glass attractors due to the asymmetry and dilution of synaptic weights, as Herz *et al.* [1987] suggested.

However, we have not known the reason of this process well, so far. Especially, the fate of many states that followed chaotic trajectories which were seen in the early stages of the evolution and the possibility to remove the still-remained-spurious-attractors are interesting to be revealed. We leave these problems to our future study.

Acknowledgments

We thank Peter Davis at Advanced Telecommunication Research Institute (ATR) for providing us great insight into the dynamics of Hopfield neural networks.

References

[Komlós and Paturi, 1988] J. Komlós, and R. Paturi, Convergence Results in an Associative Memory Model. *Neural Networks* 1, 239–250.

[Hopfield, 1982] J. J. Hopfield, Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences, USA*, 79, 2554–2558.

[Hebb, 1949] D. O. Hebb, *The Organization of Behavior.* Wiley.

[Holland, 1975] J. Holland, *Adaptation in Natural and Artificial Systems.* The University of Michigan Press.

[Goldberg, 1989] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley.

[Imada and Araki, 1995] A. Imada, and K. Araki, Mutually Connected Neural Network Can Learn Some Patterns by Means of GA. *Proceedings of the World Congress on Neural Networks*, vol.1, 803–806.

[Imada and Araki, 1997] A. Imada, and K. Araki, Evolved Asymmetry and Dilution of Random Synaptic Weights in Hopfield Network Turn a Spin-glass Phase into Associative Memory. The 2nd International Conference on Computational Intelligence and Neuroscience, *Proceedings of Joint Conference of Computer Science*, vol. 2, 223–226.

[Syswerda, 1989] G. Syswerda, Uniform Crossover in Genetic Algorithms. *Proceedings of the 3rd International Conference on Genetic Algorithms*, 2–9.

[Hertz *et al.*, 1987] J. A. Hertz, G. Grinstein, and S. A. Solla, Irreversible Spin Glasses an Neural Networks. in L. N. van Hemmen and I. Morgenstern eds. *Heidelberg Colloquium on Glassy Dynamics.* Lecture Notes in Physics, No.275, Springer-Verlag, 538–546.

[Krauth *et al.*, 1988] W. Krauth, J.-P. Nadal, and M.Mezard, The Roles of Stability and Symmetry in the Dynamics of Neural Networks. *J. Phys. A: Math. Gen.* 21, 2995–3011.