A Fate of Two Evolutionary Walkers after the Departure from the Origin

Akira Imada

Brest State Technical University
Moskowskaja 267, 224017 Brest, Republic of Belarus
ai@kk.iij4u.or.jp

Abstract. Using Sammon mappings, a method to visualize a space of multi-dimensionality, we observe a strange difference between two evolutionary searches: Evolutionary Programming and Breeder Genetic Algorithm, when they search for weight solutions of fully connected neural network model of associative memory.

1 Introduction

We have been searching for weight configurations that store a set of bipolar patterns to a fully connected neural network model as associative memory by using evolutionary computations. In usual evolutionary computations, in most cases, a search starts with a population of candidate solutions distributed at random over the search space, and those candidate solutions eventually converge to one of real solutions. Usually, we can regard the solution as a top of a peak in the fitness landscape. However, our solutions of the fully-connected neural network model of associative memory constructs *ridges*, instead of *peaks,* emanating from the origin of the search space. Our search starts with all candidate solutions being at the Origin, that is, with all weights being zero at the start. In his book, Kauffman (1993) pointed out that "*In short, the landscape is very smooth for small synaptic weights, and hence the optimum is easily sought.*" We doubt, however, this is because the origin is the cross point of all ridges of solutions, though the Origin itself is a singular point. To tackle this issue, we analyze solutions obtained by two different evolutionary searches, Evolutionary Programming (EP) and Breeder Genetic Algorithm (BGA). Solutions located by multiple runs of each of the two evolutionary computations are visualized by using Sammon mapping. And we found quite a different phenomena between EP and BGA. That is, the solutions found by EP are almost uniformly distributed, while the ones found by BGA occupies almost a similar location.

2 Associative Memory

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 information is stored by being distributed among neurons, and retrieved from dynamically relaxed neurons' states. In the model, some of the appropriate configurations of synaptic weights give the network a function of associative memory.

The Hopfield model consists of N neurons and N^2 synapses. Each neuron state is either active ($+1$) or quiescent (-1). When an arbitrary N-bit *bipolar pattern*, a sequence of $+1$ and -1, is given to the network as an initial state, the dynamical behavior of neurons' states afterwards are characterized by the strengths of the N^2 synapses. The synaptic strengths are called *weights*, and the weight from neuron j to neuron i is denoted as w_{ij} in this paper. Provided the synaptic weights are determined appropriately, network can store some number of patterns as attractors. Hopfield employed the so-called Hebb's (1949) rule to prescribe the weights.

In analyzing the Hopfield model, there have been basically two different approaches: one is to explore *pattern space* searching for attractors under a specific weight configuration, and the other is to explore *weight space* searching for an appropriate weight configuration that stores a given set of patterns. The former approach searches for the optimal pattern configurations which minimize the Hamiltonian energy *in pattern space* with the weights being fixed, while the latter searches for the weight configurations *in weight space* that optimally store a set of given fixed patterns.

In this paper, our concern is on weight space where some points give a network a capability to store a fixed set of patterns, and we call these points solutions.

3 Evolutionary Searches

Our search is started with all individuals in the first population being initialized at the origin, and we found that this is very efficient initialization compared to usual randomized one. In this simulation, a set of p random bipolar patterns ξ^ν ($\nu = 1, 2, \cdots, p$) are produced before each run. The goal is to search for a weight configuration which store all of these patterns as fixed points. Chromosome is a real-valued N^2-dimensional vector which represents a set of synaptic weights of an individual network, i.e.,

$$(w_{11}, w_{12}, \cdots, w_{1N}, w_{21}, \cdots\cdots, w_{NN}).$$

In each generation, these chromosomes are modified through evolutionary operations. Specific details of these operations are as follows.

3.1 EP

Our EP proceeds as follows. A population is constructed by μ points each of which has N^2 coordinates x_i ($i = 1, 2, \cdots, N^2$) representing weight values of a network. Each point has additional N^2 variables σ_i ($i = 1, 2, \cdots, N^2$) which are used to mutate the point itself. In this paper, all the x_i are initialized to zero and all the σ_i are initialized to 0.002. Each of these μ points is mutated by adding a Gaussian random variable of a standard deviation σ_i to each x_i. That is,

$$x_i^{\text{new}} = x_i^{\text{old}} + \sigma_i \cdot N_i(0,1).$$

This mutation produces μ mutants. After the mutation, these σ_i are also modified as:

$$\sigma_i^{\text{new}} = \sigma_i^{\text{old}} + 0.01 \cdot \sigma_i^{\text{old}} \cdot N_i(0,1).$$

We now have the original μ points and their μ mutants. The fitness value of each of the 2μ points are compared to those of q points which are chosen randomly at every time of the comparison from the whole 2μ points. Then the 2μ points are ranked according to the number of wins, and the best μ points survive (q-tournament selection). The cycle of reconstructing the new population with better fitness points and restarting the search are repeated until one of the global optima is found or a set maximum number of generation has been reached. In this paper, both λ and μ are set to 200 and q is chosen to be 10.

3.2 BGA

The BGA uses the truncation selection (Mühlenbein et al. 1995), that is, two parent chromosomes are chosen *randomly* from the best T% of the population, and mated randomly (except for itself) to produce one offspring. This is repeated until the number of offspring is equal to the size of the population.

Recombination is made with discrete crossover (Mühlenbein et al. 1995), which is similar to *uniform crossover* (Syswerda 1989) in binary chromosomes. It operates on alleles of the selected parents' chromosomes. Two parents, $(u_1, \cdots, u_n)$ and $(v_1, \cdots, v_n)$, produce an offspring $(w_1, \cdots, w_n)$ so that w_i is either u_i or v_i with equal probability.

Then an allele x_i is chosen with probability p_m to be mutated. The BGA mutation is to give a small perturbation $\Delta x_i \times \delta$ on a variable x_i, where Δx_i is a mutation range for the variable x_i, and δ is calculated as

$$\delta = \alpha_0 + \frac{1}{2}\alpha_1 + \frac{1}{2^2}\alpha_2 + \cdots + \frac{1}{2^{15}}\alpha_{15}.$$

Each α_k takes the value 1 with probability 1/16, and otherwise takes the value 0. Then mutated new allele is

$$x_i \pm \Delta_i \times \delta,$$

with the sign $+/-$ being chosen with equal probability.

3.3 Fitness evaluation:

In both search, when one of memorized patterns ξ^ν 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 network state must be similar to the input pattern ξ^ν. The similarity 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 neuron i at time t. This $m^\nu(t)$ is referred to as *overlap*. The quality of retrieval of memorized patterns are represented by $\langle m^\mu \rangle$, a temporal average of $m^\nu(t)$ over certain time interval t_0. We evaluate the fitness value of each network by further averaging these $\langle m^\mu \rangle$'s over all memorized patterns. Namely, the fitness 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 all other cases have a fitness less than 1.

4 Sammon Mapping

As Collins (1997) wrote, the dimension reduction has been an important technique for visualization of the space of high dimensionality. The Sammon Mapping (Sammon 1969) is one of these techniques. This enables us to map a set of $\mathcal{N}$ points in n-dimensional space to 2D location data so that the distance information is preserved as much as possible, or as Shine et al. (1997) wrote *"so that the n-dimensional distances are approximated by 2 dimensional distances with a minimal error."* This problem is an optimization problem.

Shine et al. (1997) and Collins (1997) proposed a method to solve this problem by a GA, as follows. First, the distance matrix whose entries are Euclidean distances between all possible pairs of $\mathcal{N}$ points in the n-dimensional space is calculated. Then tentative $\mathcal{N}$ points in the 2-dimensional space are determined representing the original $\mathcal{N}$ points in the n-dimensional space. The distance matrix of these $\mathcal{N}$ points in the 2D space is also calculated, which then will be subtracted from the original n-dimensional distance matrix, yielding an error matrix. A GA is used to minimize this error matrix.

5 Experiments

Simulations were carried out on a network with 25 neurons. A fixed set of some random bipolar patterns is given to the network to be stored as associative memory. The configurations of 25^2 weights that allow the network to store these patterns as associative memory correspond to the peaks on the landscape. They are searched for by an EP and BGA.

6 Results and Discussion

Although w_{ij} can take an arbitrary real value, two different solutions on one hyper-lines emanating from the origin are identical since they differ only by scaling factor. In other words, $\kappa \cdot w_{ij}$ works exactly in the same way as w_{ij} for any scaling factor κ. Therefore, as Gardner (1988) did, all the solutions are normalized on the surface of hyper-sphere of radius one whose center is at the origin, by regarding points on a hyper-line as the point where the line crosses the hyper-sphere surface. These normalized points are very well visualized by Summon mapping. For example Figure 1 is a mapping of 100 solutions on the $25^2 = 625$ dimensional hyper-sphere, each of which stores $p = 1$ pattern, onto two-dimensional space. It is important to note here that we made it possible by adding additional 100 random points very close to the origin in order to maintain the difference relation between the points in the 625-dimensionality.

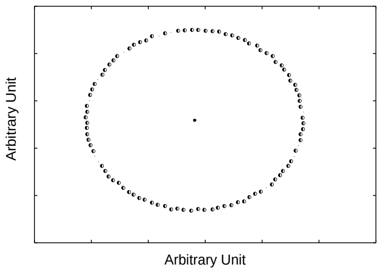

Fig. 1. Sammon mapping of 100 solutions ($p = 1$) located by EP.

Then a pair of 30 different solutions for $p = 6$ located by 30 runs of EP and BGA, respectively, are visualized. The results are shown in Figure 2.

Mühlenbein et al. (Mühlenbein et al. 1995), concluded that *"the BGA mutation scheme is able to optimize many multi-modal functions."* However our observation was opposite, that is, BGA converges always to a similar solution while EP converges to different solutions for each run. We also started BGA with a randomly distributed first generation, instead of all zero population, but the results were the same. They always converged to one solution.

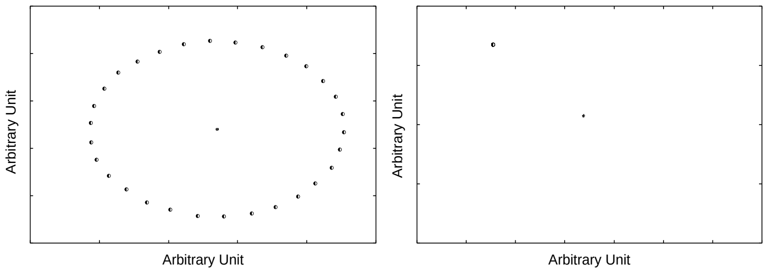

Fig. 2. Sammon mapping of 30 solutions ($p = 6$) located by EP (left) and BGA (right)

7 Conclusion

We have observed solutions of the fully connected neural network model those located by two different evolutionary searches, EP and BGA, and found that multiple runs of EP locates multiple solutions while BGA locates a single solution out of them. The reason awaits our further analysis.

Reference

Collins, T. D. (1997) *Using Software Visualization Technology to Help Evolutionary Algorithm Users Validate their Solutions.* Proceedings of the 7th International Conference on Genetic Algorithms, pp.307–314.

Gardner, E. (1988) *The Phase Space of Interactions in Neural Network Models.* Journal of Physics, 21A, pp.257-270.

Hopfield, J. J. (1982) *Neural Networks and Physical Systems with Emergent Collective Computational Abilities.* Proc. of the National Academy of Sciences, USA 79, pp.2554-2558.

Hebb, D. O. (1949) *The Organization of Behavior.* Wiley.

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

Mühlenbein, H. and D. Schlierkamp-Voosen (1995) *Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization.* Evolutionary Computation Vol.1 (1996), pp.25–49.

Sammon, J. W. (1969) *A Nonlinear Mapping for Data Structure Analysis.* IEEE Transactions on Computers. C-18(5), pp.401–408.

Shine, W. B. and C. F. Eick (1997) *Visualizing the Evolution of Genetic Algorithm Search Processes.* Proceedings of the IEEE International Conference on Evolutionary Computation, pp.367–372.

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

This article was processed using the LaTeX macro package with LLNCS style