A Learning Machine That Evolves

Kaoru NAKANO, Hideaki HIRAKI and Shiro IKEDA
Department of Mathematical Engineering and Information Physics
Faculty of Engineering, University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113, Japan
E-mail: hiraki@bcl.t.u-tokyo.ac.jp

ABSTRACT

In this article we propose a simple model of a learning machine that evolves. When a classification problem is given, a Perceptron-like learning machine obtains a proper set of feature-detecting cells through mating, mutation, and natural selection. Computer simulation showed the expected results. This is one of our trials to approach the evolutionary system in the real world.

1. Introduction

This is a "synthetic approach" to the evolutionary system. If a model is constructed whose behavior is very similar to that of the evolutionary system in the real world, the essential mechanism of the system will be estimated from the model. We call it the "synthetic approach." In the approach it is desirable to construct a complete model of the evolutionary system, but this is difficult, so we accumulate trials to construct models for partial mechanisms of the system.

According to this idea, we have proposed a model for self-organization, where various robots are automatically produced from several kinds of elements through mating and mutations[2]. In the model, it is presumed that if a structure of the elements (robot) is given, it will have a specific ability. There must be structure information in order for self-organization to occur. We call it a "system description." We devised the system descriptions such that two of them are mated and are affected by mutations into a new system description which describes a new able robot. We did not consider natural selection in that model.

Now, in this article we don't focus on the system description. In self-organization, it is also important to study the mechanism of evolution. Practically we applied the method to Perceptron-like learning machines. As explained in the next section, a Perceptron can learn to categorize patterns if it has suitable set of feature-detecting cells[1]. It is hard to choose the right set of feature-detecting cells. Back-propagating error-correcting method is effective for the case. Hidden cells become suitable for the given classification problem (the task) through learning. But the method has two demerits. One is that the learning not always converges to the optimal point but is trapped in local minima. The other is that, usually, the number of the hidden cells needed for the given problem has to be determined beforehand.

Our evolutionary method was applied to that matter instead of the backpropagation. As a result of it, both of the number of the feature-detecting cells and the function of each cells became suitable although evolutionary computation took a lot of time.

2. A Perceptron and Back-Propagating Method

A Perceptron is a learning machine with a neural network structure proposed by F. Rosenblatt[3] and studied well by M. Minsky and S. Papert[1]. It learns to judge whether a given pattern belongs to a certain category or not. A typical structure of a Perceptron consists of an input layer, a hidden layer and an output cell as shown in Fig.1. Patterns are shown to the input layer. A cell in the hidden layer is connected to some cells in the input layer (they form the "receptive field" of the cell) and extracts a feature of the given pattern. If the corresponding feature is detected by the cell (suppose that black points are 1's and white points are 0's), it outputs 1, otherwise 0. The output cell is connected to all the cells in the hidden layer, and outputs 1 if the weighted sum of their outputs is larger than the threshold value. Therefore the output is considered to be a logical function of the outputs of the input layer and also of the outputs of the hidden layer. Training is to change the weights and the threshold value whenever the judgement is wrong. Although explanation of the detail of the training is omitted, the learning is proved to converge in finite iterations if the training method is appropriate and the

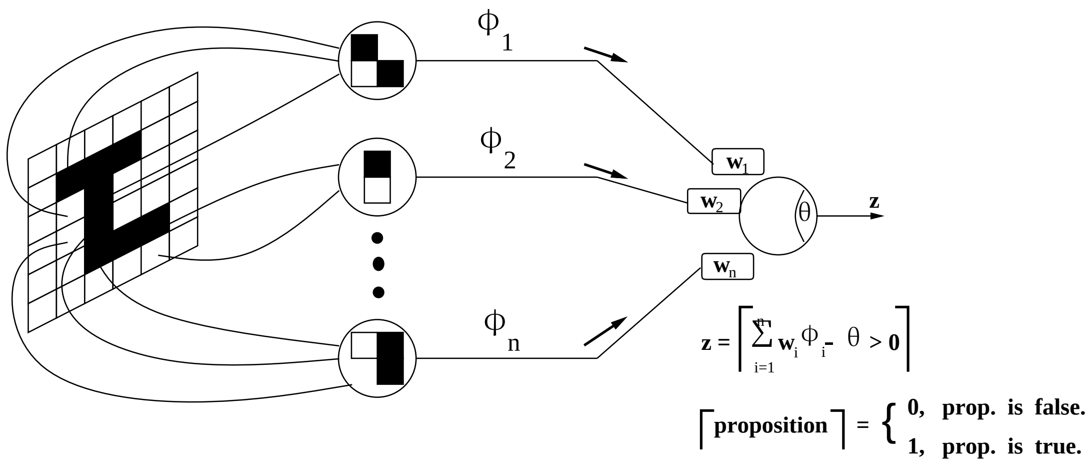

Fig. 1 Perceptron. (The same feature-detecting cells are assumed to be located in parallel at every place of the layer.)

logical function of the outputs of the hidden layer is linearly-separable.

By M. Minsky and S. Papert, every logical function has a positive normal form. It means there exists a set of mask-type feature-detecting cells that can make an arbitrary logical function linearly-separable. The mask-type feature-detecting cell means that the output of the cell is a conjunction function (AND gate) of the inputs. Our evolutionary algorithm is based on the idea of searching over this kind of cells.

A Perceptron can learn any classification of patterns if there are feature-detecting cells (hidden layer cells) enough to make the task linearly-separable. But the learning is done only on the output cell. Enormous feature-detecting cells must be prepared if they are chosen at random. The generalized delta rule and back-propagating-error method were proposed to solve the problem[4], where hidden layer cells can learn according to the steepest gradient method. But there is no general method to estimate how many hidden layer cells are needed. Moreover, it is known that convergence of the learning is often trapped in local minima.

Instead of using the backpropagation we apply our evolutionary algorithm to this problem as explained in the next section.

3. Evolutionary Algorithm

In this section, we describe the evolutionary algorithm that is applied to a Perceptron. Details of the process are described in Examples 1 and 2 in the next section.

In evolution of living things, we think that simple cells gather to form a structure with complicated functions. In our model, evolution begins from simple feature-detecting cells that detect the brightness on a cell in the input layer. Individuals (learning machines) obtain feature-detecting cells with larger receptive fields through generations. In the mutation described below, feature-detecting cells generated at random are added to individuals. Each of the feature-detecting cells will have a receptive field consisting of arbitrary number of points which are selected to be black or white at random. Each pattern in the receptive field corresponds to the feature of each cell. Examples of the features with adjacent points are shown in Fig.2. Generally, the points do not need to be adjacent.

In order to recognize complicated patterns, a set of many kinds of feature-detecting cells will be necessary. By the mating described below, increasing and decreasing the number of the feature-detecting cells are realized. Using our evolutionary algorithm, a reproduction system is constructed. Each generation includes a population of individuals. From the first generation, two individuals are chosen and are mated (sometimes with mutations) into an individual of the next generation. The process is repeated until the necessary population of the next generation is attained. In the same manner the reproduction process proceeds to further generations. If natural selection is added to this process, evolution will be realized.

3.1. Mating

Two individuals (parents) are mated into an individual (child) in the reproduction process. In the process partial information of the two system descriptions of the parents are exchanged. Generally, a system description describes the structure of an individual. In this case a system description is a set of feature-detecting cells itself. As the mating, we devised the "mixing pot method" (see Fig.3).

The sets of feature-detecting cells of the parents A and B are in pot A and pot B respectively. We

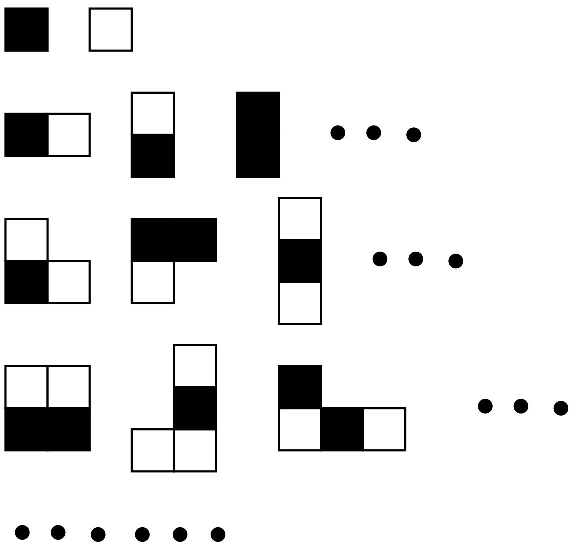

Fig. 2 Receptive fields and functions of feature-detecting cells.

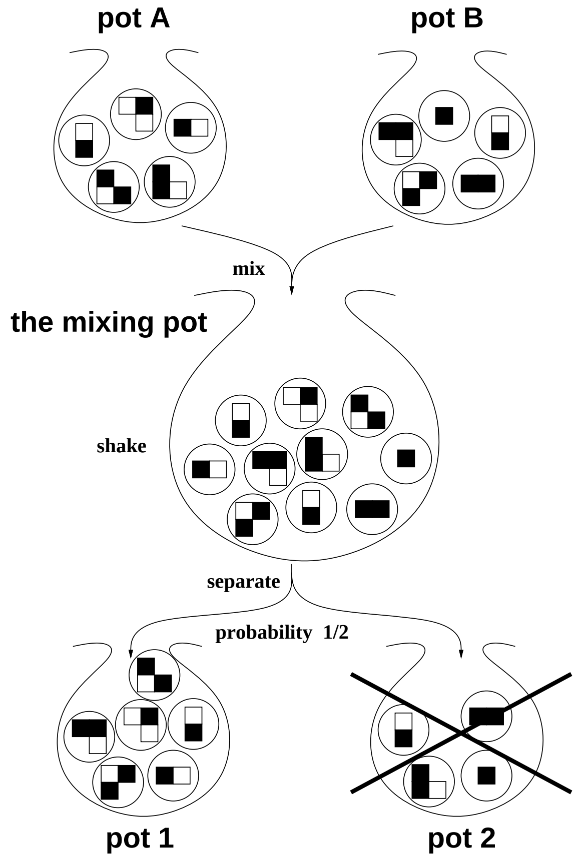

Fig. 3 The mixing pot method.

pour the contents into a mixing pot and shake it well. Then a cell is taken out and put into pot 1 or pot 2 with equal probability $p = 1/2$. The process is repeated until no cell is in the mixing pot. It is not permitted for the cells with same features to be put into one pot. In such cases, they are separated into pot 1 and pot 2. Pot 2 is discarded and pot 1 is given to the child. By this method, number of feature-detecting cells of an individual can gradually increase and decrease. If natural selection is added to this method, it will be expected that minimum number of feature-detecting cells necessary for the given task are attained.

3.2. Mutation

The mutation is also necessary for the evolution. Without it, the variety of feature-detecting cells would be limited. In our model mutations play the role to introduce new feature-detecting cells into the system. A mutation occurs at random with a certain probability. When it occurs, a feature-detecting cell is added to a learning machine, whose receptive field and function are chosen at random. So there is the possibility that individuals obtain complicated feature-detecting cells with large receptive fields. We assumed that larger receptive fields have less probability to be chosen.

3.3. Natural Selection

When a new generation is produced, each individual learns patterns certain learning times. Then a score of perception is counted. The score is defined to be the ratio of correct answer divided by the number of the feature-detecting cells. The higher the score of an individual is, the more probable it is that the individual is selected as a parent. We permit the same parents to be selected. By this method, we expect the child to reflect the characters of the parents. Consequently, through generations, the child is expected to obtain the set of feature-detecting cells with suitable number and functions.

4. Computer Simulation

We made simple experiments to ensure that the evolutionary algorithm works well. The tasks are to distinguish black-and-white patterns on 6 times 6 pixel screen. For simplicity, it is assumed that every receptive field consists of less than 4 adjacent

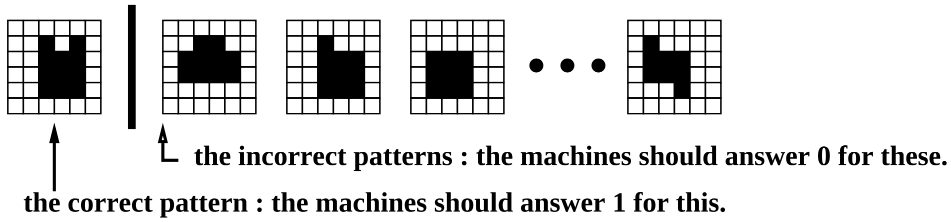

Fig. 4 The given task in example 1.

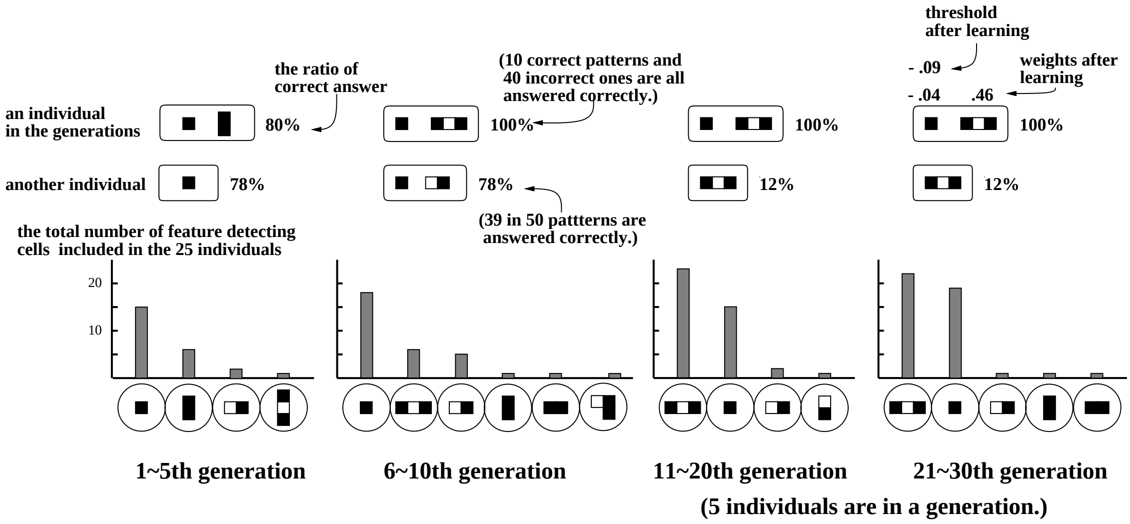

Fig. 5 The results of example 1.

cells in the input layer (the upper 3 rows in Fig.2). The first example is to distinguish a certain pattern "horns" from the others. The second is to distinguish patterns in the category "hollow rectangle" from the others.

4.1. Example 1

See Fig.4. We let the learning machines try to distinguish the left pattern (correct pattern) from the others. The location of the pattern is arbitrary. It is assumed that the same feature-detecting cells are located in parallel at every place of the layer. Consequently, when a feature-detecting cell is added in mating and mutation, it is applied to the entire input layer. Presume that in each generation there are 5 individuals. After all the individuals in a generation have been produced, each individual learns 50 patterns 50 times. Those patterns contain 10 correct patterns (all are the same) and 40 incorrect ones. Ratio of correct answer is calculated for these 50 patterns after the learning. We assume that in mutations, probability of selecting cells with the receptive field of 1 point is 0.5, 2 points 0.3 and 3 points 0.2. In the first generation we assume that every individual has no feature-detecting cells. Also we assume the occurring probability of a mutation (in reproduction of an individual) is 0.3.

The results of the experiment are in Fig.5. The graphs show number of each kind of feature-detecting cells through all the individuals in each 5 generations (25 individuals). Above each graph, two examples are shown with their feature-detecting cells and with the ratio of correct answer.

We can see that individuals have obtained the suitable sets of feature-detecting cells. At the early stage of generations, they obtained feature-detecting cells with small receptive fields (1 or 2 points). The individuals tried to calculate the number of black points because the correct pattern is just one fixed pattern. As the reproduction advanced, each individual obtained a feature-detecting cell sensitive to three adjacent points both sides of which are black. With the cell it can recognize the horns and the ratio of correct answer rose to 100%. The evolution did not completely converge because mutations occurred. If we gradually decrease the occurring probability of mutations, it will converge. But it is not essential to the evolution. It is also seen that the individuals with feature-detecting cells at the left side of the graph (frequency is high) have high ratio of correct answer. The feature-detecting cells at the right side of the graph (frequency is low) come in and go out of the system by mutations through generations.

4.2. Example 2

See Fig.6. We let the individual learning machines try to distinguish hollow rectangles (for example, the upper row in the figure) from the other patterns (the lower row). In this example, we assume that a generation includes 10 individuals and that there are 50 patterns containing 10 correct patterns and 40 incorrect ones. The results shown in Fig.7 are given by summing up the individuals of every 10 generations. Other parameters are the same as Example 1. In this case correct patterns form a category, so the task is more difficult than Example 1. The individuals obtained more feature-detecting cells.

5. Conclusions

Evolutionary algorithm was applied to Perceptron-like machines. Although the algorithm is very simple, it is flexible. The machine automatically got a set of cells with suitable number and functions.

Evolutionary computation needs much computing time and it does not converge because mutations occur. That means evolutionary system is always dynamic and can adapt to the change of the circumstance. We believe that the essential mechanism of evolution was shown in our trials. Through this kind of trials, we are trying to elucidate evolutionary system.

We express our hearty thanks to Katsumi KONISHI, Hiroki YOKOYAMA and Shingo UCHIHASHI for their assistance.

References

[1] M. Minsky and S. Papert, *Perceptrons, Expanded Edition.* MIT Press, 1988.

[2] K. Nakano, S. Uchihashi, N. Umemoto, and H. Nakagama, “An approach to evolutional system,” in *ICEC’94*, pp. 781–786, 1994.

[3] F. Rosenblatt, *Principles of Neurodynamics, perceptrons and the theory of brain mechanisms.* Spartan, 1961.

[4] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, pp. 533–536, 1986.

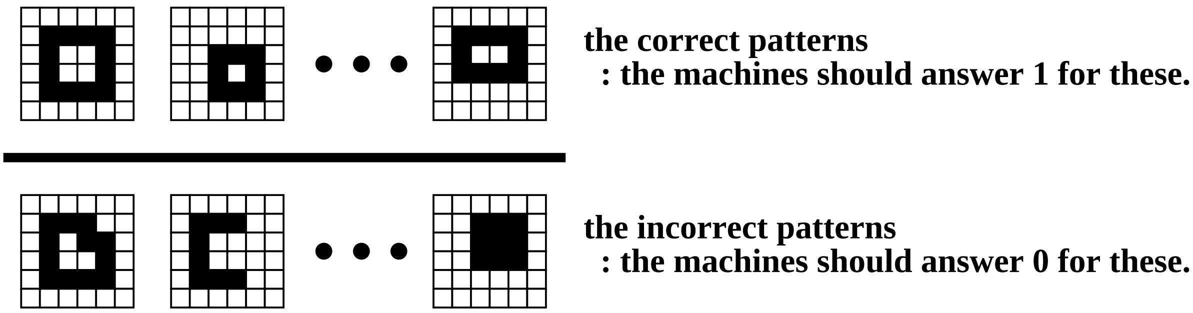

Fig. 6 The given task in example 2. (Discriminating "hollow rectangles" from the other patterns.)

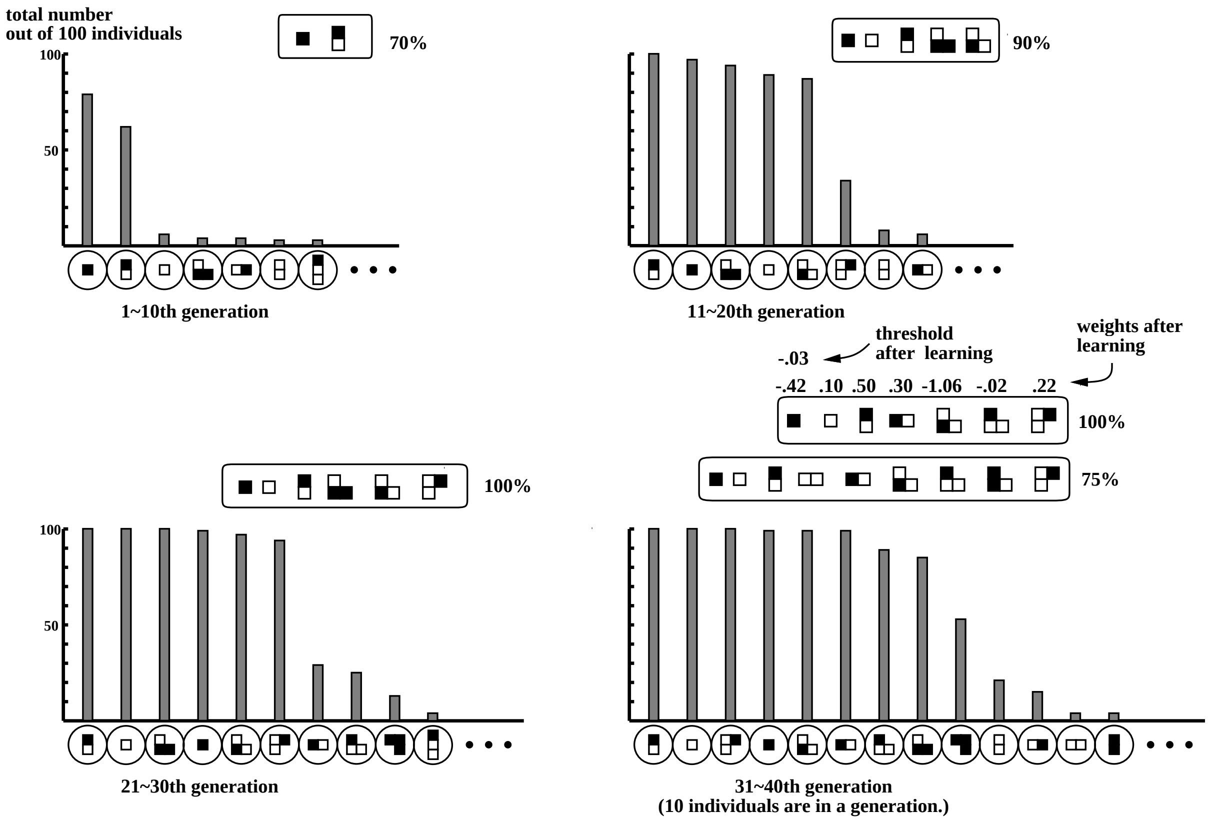

Fig. 7 The results of example 2.