Batch Learning of the Self-Organizing Relationship (SOR) Network

Takeshi Yamakawa, Keiichi Horio and Satoshi Sonoh
Graduate School of Life Science and Systems Engineering
Kyushu Institute of Technology
2-4 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka, 808-0196, Japan
E-mail: yamakawa@brain.kyutech.ac.jp, horio@brain.kyutech.ac.jp, sonoo-satoshi@edu.brain.kyutech.ac.jp

(Submitted on June 4, 2005)

Abstract - In this paper, we propose a batch learning of the SOR network which extracts a desirable I/O relationship using “Attractive Learning” and “Repulsive Learning”. The SOR network takes advantage of fast learning and less number of parameters similar to the batch version SOM. The effectiveness of the proposed algorithm is verified by simple experiments.

Keywords - Batch Learning, Self-Organizing Relationship (SOR) Network, Learning Vector with Evaluation

1. Introduction

A Self-Organizing Relationship (SOR) network [1] which is an extension of a Self-Organizing Map (SOM) [2] is a very useful tool for modeling of a desirable I/O relationship of an unknown system. In the SOR network, the I/O vector pairs with their evaluations are used as learning vectors and the desirable I/O relationship is extracted from the learning vectors as weight vectors. In the learning process, the weight vectors are updated to come close to or to get away from a learning vector when its evaluation is good or bad, respectively. The SOR network has been successfully applied to a modeling of nonlinear systems [3], a design of trailer-truck back-up controller [4] and so on. In the conventional learning algorithm of the SOR network, the learning result depends on the learning parameters and the order of applying the learning vectors. In SOM, it is known that a batch learning algorithm [5] [6] [7] is effective for solving these problems. However, the batch learning algorithm of the SOM can not be applied to the SOR network as it is, because an updating procedure is different in good and bad evaluations of the learning vectors.

In this paper, we propose the batch learning of the SOR network which is independent of the order of applying the learning vector and a specification of the learning parameters. In the batch learning algorithm, the learning vectors, which have bad evaluations and cause the repulsive force to the weight vectors, are converted to new learning vectors. The converted learning vectors are regarded as virtual learning vectors which have good evaluations to force the weight vectors to the opposite direction. The SOR network is updated in a similar manner to batch version SOM using the converted new learning vectors. The proposed batch learning of the SOR network can extract the desirable I/O relationship faster. The effectiveness of the proposed batch learning algorithm is verified by some experiments.

2. Self-Organizing Relationship Network

The SOR network consists of the input layer, the output layer, and one or two dimensional competitive layer, in which \( n \), \( m \), and \( N \) units are included, respectively, as shown in Figure 1. The \( i \)-th unit in the competitive layer is fully connected with the input and output layers through weight vector \( v_i \).

\[
v_i = (w_i, u_i),
\]

where, \( w_i \) is the weight vectors between the input layer and the competitive layer, and \( u_i \) is the weight vectors between the output layer and the competitive layer.
In the SOR network, both I/O vector pairs and their evaluations for a desirable I/O relationship are employed as learning vectors. In the SOM, only input topology is mapped from a high dimensional input space to one or two dimensional competitive layer. On the other hand, in the SOR network, an I/O topology is mapped from a high dimensional I/O space to the competitive layer. An original feature of the SOR network is that the weight vector updating is determined by an evaluation. The evaluation is designed on the basis of a relation between the learning vectors and a desirable I/O relationship. If the evaluation is a good or positive value, the weight vectors are updated to come close to the learning vector. This operation is called “attractive learning”. On the other hand, if the evaluation is a bad or negative value, the weight vectors are updated to get away from the learning vector. This operation is called “repulsive learning”. The desirable I/O relationship is extracted by the attractive learning and the repulsive learning.

The incremental learning (conventional algorithm) of the SOR network is processed as follows.

Step 1 One I/O vector pair \( I_l = \{x_l, y_l\} \) is randomly selected from the \( L \) I/O vector pair set, and applied to the input and the output layers as a learning vector.

Step 2 A winner unit \( l^* \) for the learning vector \( I_l \) is selected by the smallest Euclidean Distance by following equation:

\[
l^* = \arg \min_i \| I_l - v_i \|.
\]

Step 3 A coefficient \( h_{i,l} \) of neighboring effect is calculated by following neighboring function:

\[
h_{i,l} = |E_l| \exp\left( -\frac{\| p_i - p_{l^*} \|^2}{2\sigma(t)^2} \right),
\]

where \( E_l \) is an evaluation of the learning vector \( I_l \). \( p_i \) and \( p_{l^*} \) are positions of the \( i \)-th unit and the winner unit on the competitive layer, respectively. \( \sigma(t) \) is a width of neighboring function at learning step \( t \).

Step 4 The weight vectors are updated by the following equation.

\[
v_i^{new} = \begin{cases} v_i + \alpha(t)h_{i,l}(I_l - v_i) & (\text{for } 0 \leq E_l) \\ v_i - \beta(t)h_{i,l}\exp(-\| I_l - v_i \|)\sgn(I_l - v_i) & (\text{for } E_l < 0), \end{cases}
\]

where \( v_i^{new} \) is the weight vector after updating. \( \alpha(t), \beta(t) \) are learning rates for attractive or repulsive learning at learning step \( t \), respectively.

Step 5 Steps 1 to 4 are repeated decreasing the learning rates \( \alpha(t) \) and \( \beta(t) \) and the width of neighboring function \( \sigma(t) \).

It is assumed that a cluster of bad input data is presented sequentially. In this case, the weight vectors are moved far away from the desirable I/O relationship, and are often diverged. Thus, the SOR network may not be able to extract a desirable I/O relationship. In addition, the cumbersome problems in the incremental learning are summarized as follows.

32
Figure 2. The learning vectors with positive evaluations (○) cause the attractive learning and a learning vector with a negative evaluation (●) causes the repulsive learning. The learning vector $I_1$ with the negative evaluation is converted to the virtual attractive learning vector $I_1'$ (a). If a weight vector (×) is near to the learning vector with the negative evaluation, a repulsion coefficient is big (b). The weight vector is updated toward a center of gravity of all the learning vectors after the conversion (c).

1. A learning result depends on an order of applying the learning vectors.

2. Assignment of reasonable learning parameters $\alpha(t)$ and $\beta(t)$ is difficult.

3. A convergence speed is slow.

3. Batch Learning of the SOR Network

In this paper, we propose the batch learning for the SOR network to overcome the problems mentioned above. The batch learning algorithm of the SOM can not be applied to the SOR network as it is, because a center of gravity of learning vectors can not be solved due to coexistence of attraction with repulsion. In the proposed batch learning algorithm, the learning vectors which cause repulsive learning are converted to virtual attractive learning vectors by the following equation.

$$I_i' = \begin{cases} I_i & \text{(for } 0 \leq E) \\ v_i + r \frac{v_i - I_i}{\|v_i - I_i\|} & \text{(for } E < 0) \end{cases}$$

where $r$ is a repulsion coefficient, it is represented by

$$r = r_0 \exp\left( - \frac{\|v_i - I_i\|^2}{2\sigma_r^2} \right),$$

where $r_0$ is a deviation from the weight when a learning vector hits to the weight vector, $\sigma_r$ is the width of repulsion effect. The converted learning vector is called a “virtual learning vector”. The advantage of this conversion is that the evaluation of the converted learning vector is able to be regarded as a positive evaluation. By taking this conversion, the batch learning algorithm similar to a batch version SOM can be applied to the SOR network.

Figure 2 shows some examples of the conversion. If a weight vector is near to the learning vector with the negative evaluation, the repulsion coefficient $r$ is large to realize a strong repulsive learning. The weight vector is updated toward a center of gravity of learning vectors after conversion (shaded triangle) as shown in Figure 2 (c).

Batch learning of the SOR network is illustrated in the following.

**Step 1** One I/O vector pair $I_l = \{x_l, y_l\}$ in the $L$ I/O vector pair set is applied to the input and output layers as a learning vector.
Figure 3. The clusters of learning vectors with positive (opened squares) and negative (filled squares) evaluations (a). The mean errors corresponding to learning parameters for the incremental learning (b) and the proposed batch learning (c). (i), (ii), (iii), (iv) and (v) correspond to those in Figure 4.

Step 2 A winner unit $l^*$ for the learning vector $I_l$ is selected by the smallest Euclidean Distance by the following equation:

$$l^* = \arg \min_i \| I_l - v_i \|.$$  \hspace{1cm} (7)

Step 3 A coefficient $h_{i,l}$ of neighboring effect is calculated by the following neighborhood function:

$$h_{i,l} = |E_l| \exp\left( -\frac{\| p_i - p_{l^*} \|^2}{2\sigma(t)^2} \right),$$  \hspace{1cm} (8)

where $p_i$ are $p_{l^*}$ are positions of the $i$-th unit and the winner unit on the competitive layer, respectively. $\sigma(t)$ is a width of neighboring effect.

Step 4 After applying all the I/O vector pairs, the weight vectors are updated by using the converted learning vectors $I_l'$ in Eq. (5).

$$v_{i}^{new} = \frac{\sum_{l=1}^{L} h_{i,l} I_l'}{\sum_{l=1}^{L} h_{i,l}}$$  \hspace{1cm} (9)

where $v_{i}^{new}$ is the weight vector after updating.

Step 5 Steps 1 to 4 are repeated decreasing the width of neighboring effect $\sigma(t)$.

In the proposed batch learning, the weight vectors are updated independently of the order of applying the learning vectors. In addition, the convergence speed is significantly faster.

4. Experimental Results

In order to verify the effectiveness of the proposed batch learning, the following three experiments are achieved.

4.1 A designing of the learning parameters

A relationship between learning parameters and the learning results is examined as follows. I/O vector pairs as learning vectors are randomly generated inside the open squares or the filled squares shown in Figure 3 (a).
Figure 4. Five weight vectors (×) and their neighborhood relation shown in the line. In case that the repulsive learning parameter for the incremental learning is zero (i), small (ii) and large (iii), it is often diverged. In case that the repulsive learning parameter for the proposed batch learning is small (iv) and large (v), it is very stable. Here, five cases are illustrated in Figure 3 (b) and (c).

The learning vectors are given from the open and the filled squares with the positive ($E = +1$) and the negative ($E = -1$) evaluations, respectively. 100 learning vectors of each square are applied for the experiment.

Here, we introduce the error function of the SOR network to evaluate the map acquired by the learning. The error function of weight vector $w_i$ is represented as follows:

$$
\text{error}_i = \frac{1}{2} \sum_{l \in P} \sum_{j=1}^{N} h_{i,l} \| I_l - v_j \|^2 + \sum_{l \in M} \sum_{j=1}^{N} h_{i,l} r_e \exp(-\frac{\| I_l - v_j \|^2}{2\sigma^2}),
$$

where $P$ and $M$ are a set of learning vectors with positive or negative evaluations, respectively. $r_e$ is $r$ is the repulsion coefficient for the error function. Since the error function is described with coefficients $h_{i,l}$, it is also done with evaluation values $E_l$. The first term means the vector quantization error of the learning vectors with positive evaluations, and the second term means the avoidance error of those with negative evaluations. This error function is similar to the one of SOM [8] excluding the effect of the repulsion.

In the incremental learning, the mean error (of ten trials) changes with learning parameter ratio ($\beta/\alpha$) with its standard deviation as shown in Figure 3 (b). The result is very sensitive to the learning parameter ratio. The conditions of errors can be analyzed by the map after the learning. In case that the repulsive learning is not done at all, the topological map is constructed without an avoidance of the learning vector with negative evaluations as shown in Figure 4 (i). In case of small repulsive learning rate, the topological map is just constructed to avoid the areas where the negative evaluations exist as shown in Figure 4 (ii). However, when large repulsive learning parameters, the learning turns out a failure as shown in Figure 4 (iii), and thus the error grows larger. Furthermore, the reasonable learning parameters should be assigned in accordance with each set of learning vectors.

In contrast, the mean error corresponding to learning parameters in the proposed batch learning is smaller than that in the incremental learning as shown in Figure 3 (b) and (c). The avoidance error when the repulsive learning parameter is small became large as shown in Figure 4 (iv). In case that the parameter is large, the topological map is accurately constructed with small avoidance error as shown in Figure 4 (v). The large repulsive learning parameter makes the error a stable value because the weight vectors are updated far away as the repulsion can not affect those. This results show that a designing of the learning parameters became easy by the proposed batch learning.

4.2 Modeling nonlinear function and its convergence speed

In the execution mode of the SOR network, a desirable I/O relationship is modeled using the weight vectors after the learning. The output vector $\hat{y}$ for test input vector $\hat{x}$ is generated according to following equations.

$$
\hat{y} = \frac{\sum_{i=1}^{N} z_i u_i}{\sum_{i=1}^{N} z_i}
$$
where $z_i$ is similarity between the test input vector $\hat{x}$ and the weight vector $w_i$ calculated by:

$$z_i = \exp\left(-\frac{\|\hat{x} - w_i\|^2}{2\gamma^2}\right),$$

(12)

where $\gamma$ is a coefficient of the similarity.

The performance of the proposed batch learning is evaluated by a convergence speed of the learning and a modeling accuracy of an I/O relationship. Figure 5 (a) shows 1000 learning vectors that are generated from uniform random numbers $[0 : 1]$ and their evaluations based on following nonlinear I/O surface,

$$y_d = \frac{2 + \sin(2\pi x_1) + \cos(2\pi x_2)}{4}.$$  

(13)

The evaluations $E_l$ are defined by the following:

$$E_l = 1 - 10d_l,$$

(14)

where, $d_l$ is a distance between a desirable output vector $y_d$ and the generated output vector.

The proposed batch learning of the SOR network which has 100 units ($10 \times 10$) is applied to extract the I/O relationship from the learning vectors. A strength of the repulsion effect $r_0$ and its width $\sigma_r$ are 1 and 0.02, respectively. $\sigma$ is exponentially decreased from initial value $5\sqrt{2}$ to 0. The distribution of the weight vectors after 30 learning steps is represented Figure 5 (b). Figure 5 (c) shows a curved surface that is modeled by the execution mode. The output vectors are calculated by 2000 test input vectors that are not used in the learning. **Root-Mean-Square-Error** (RMSE) of the output vectors which are generated by the execution mode and the desirable output vector is 0.035.

A modeling accuracy of the proposed batch learning and a calculation cost are evaluated by comparing with on-line learning. Figure 6 shows the RMSE corresponding to CPU time necessary for the learning. In the on-line learning, over $10^3$ seconds (as 1000 learning steps) are required for the convergence. On the other hand, in the batch learning, only 10 seconds (as 50 learning steps) are required for the convergence. The convergence speed of the proposed batch learning is 1000 times or more faster than on-line learning. Furthermore, the proposed batch learning holds a better modeling accuracy.

5. Conclusion

The SOR network has the important function that takes the advantage of the attractive and repulsive learning based on the evaluation value for the acquisition of the I/O relationship. But, in the incremental learning, the SOR network has problems that (1) the learning result depends on an order of applying the learning vectors and that (2) the convergence speed of the learning is slow because the learning vectors which have positive evaluations or negative evaluations are applied successively to the network. Furthermore, it is necessary to (3) assign the reasonable learning parameter by trial and error. These problems are caused by the coexistence of attractive learning and repulsive learning.

---

**Figure 5.** The desirable output vectors are generated by the execution mode of the SOR network. 500 learning vectors with the positive evaluations (○) and 500 learning vector with the negative evaluations (●) (a). The distribution of the weight vectors after batch learning (b). The curve surface generated by the execution mode (c).
Our purpose is to expand the usability and application range of the SOR network. In this paper, the virtual attractive learning vector, which was obtained by conversion of the repulsive learning vector, was introduced for the batch learning of the SOR network. The batch learning of the SOR network which enables that the weight vectors are updated toward the center of gravity of the learning vectors was achieved by the virtual attractive learning vectors.

It is confirmed that the proposed batch learning is effective to both the assignment of the learning parameters and the acceleration of the convergence speed, by some experiments. In particular, the convergence speed became 1000 times or more faster than the conventional learning. It facilitates the SOR network to be used in the real-time application.

Acknowledgment: This work was partially supported by the 21th Century COE (Center of Excellence) Program in Kyushu Institute of Technology, entitled “World of brain computing interwoven out of animals and robots”, and was also supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (A), no. 14205038, 2005.

References

Takeshi Yamakawa was born in 1946. He received the B.E. degree in electronics engineering in 1969 from Kyushu Institute of Technology and the M.E. degree in electronics engineering in 1971 from Tohoku University, both in Japan. He received the Ph.D. degree for his studies on electrochemical devices in 1974 from Tohoku University, Japan. He was an Associate Professor at Kumamoto University. He joined the faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology (KIT), Iizuka, Japan in April 1989 as a full professor. He is a professor of the Graduate School of Life Science and Systems Engineering, KIT since 2001. He is also the chairman of Fuzzy Logic Systems Institute (FLSI), the foundation established by himself in 1990. His main research interest lies on hardware implementations of fuzzy systems, fuzzy neural networks, and chaotic systems, and also on the micro machining and the micro-electrophoresis device. He holds 11 patents in U.S.A., 4 patents in Europe, 1 patent in Australia and 1 patent in Taiwan, and he has also applied for more than 80 patents in Japan. He received the International MOISIL Prize and the Gold Medal in the Fuzzy Systems Engineering in 1994, Distinguished Service Prize from Biomedical Fuzzy Systems Association in 1998, Doctorate Honoris Causa from University of Pitesti, Romania in 1999, Lifetime Achievement Award from WAC in 2000, Gabor Award from IJCNN 2001, Fellowship Award from Japan Society for Fuzzy Theory and Intelligent Informatics in 2003. Dr. Yamakawa is a member of IFSA and other 9 academic institutes. He is acting as a member of editorial board and a regional editor of 19 international professional journals. He contributes more than 50 international conferences as an organizer or a member of organizing/programming committee.

Keiichi Horio was born in 1972. He received the B.E., the M.E. and the Ph.D. degrees from Kyushu Institute of Technology (KIT), Japan in 1996, 1998 and 2001, respectively. He is an assistant professor of the Graduate School of Life Science and Systems Engineering, KIT, Japan since 2003. He was a Research Fellow of Japan Society for Promotion of Science at KIT. He is a member of IEICE and Japan Society for Fuzzy Theory and Intelligent Informatics (SOFT).

Satoshi Sonoh was born in 1982. He received the B.E. degrees from Kyushu Institute of Technology (KIT), Japan in 2004. He is now being educated, as a COE student, in the Multi-disciplinary Education program. His research interest lies on the hardware implementation of self-organizing systems. He is a member of Japanese Neural Network Society (JNNS) and Japan Society for Fuzzy Theory and Intelligent Informatics (SOFT).