IEICE TRANS. FUNDAMENTALS, VOL.E99–A, NO.3 MARCH 2016

PAPER

Multi-Layer Perceptron with Pulse Glial Chain

Chihiro IKUTA[†a)], ***Nonmember***, **Yoko UWATE**[†b)], **Yoshifumi NISHIO**[†c)], ***Members***, ***and*** **Guoan YANG**[††d)], ***Nonmember***

SUMMARY Glial cells include several types of cells such as astrocytes, and oligodendrocytes apart from the neurons in the brain. In particular, astrocytes are known to be important in higher brain function and are therefore sometimes simply called glial cells. An astrocyte can transmit signals to other astrocytes and neurons using ion concentrations. Thus, we expect that the functions of an astrocyte can be applied to an artificial neural network. In this study, we propose a multi-layer perceptron (MLP) with a pulse glial chain. The proposed MLP contains glia (astrocytes) in a hidden layer. The glia are connected to neurons and are excited by the outputs of the neurons. The excited glia generate pulses that affect the excitation thresholds of the neurons and their neighboring glia. The glial network provides a type of positional relationship between the neurons in the hidden layer, which can enhance the performance of MLP learning. We confirm through computer simulations that the proposed MLP has better learning performance than a conventional MLP.

key words: *multi-layer perceptron, glial chain, classification*

1. Introduction

The adult human brain and the rest of the central nervous system comprise up to one trillion nerve cells, including excitable nerve cells and synapses. The cells in the central nervous system are classified into two types, namely, neurons and glia, including astrocytes and oligodendrocytes. Many studies have investigated the biological features of neurons and their functions. Neurons can transmit and gather electrochemical signals from each other, thereby accomplishing such brain tasks as thinking and memory. So far the astrocytes of the glia had not been investigated deeply because the functions of astrocytes in the brain were difficult to investigate. Hence, these cells were long considered to be merely support cells for the neurons. Recently, researchers discovered that an astrocyte can transmit signals by adjusting the concentration of ions in the glia [1]–[3]. In addition, an astrocyte has many receptors for ions such as adenosine triphosphate (ATP), glutamic acid (Glu), and calcium ions (Ca^{2+}). These ions are important for brain function because neurons also use ATP and Glu in synapses, and astrocytes generates Ca^{2+} concentration wave [4], [5]. Among them, we have focused on Ca^{2+} because astrocytes induce ATP, D-serine, etc. in accordance with changes in Ca^{2+} concentration [6]–[9]. ATP and D-serine directly influence the membrane potentials of neurons. A Ca^{2+} concentration wave is transmitted to the astrocytes in a wide range of the brain. The neurons are also influenced by the Ca^{2+} concentration wave through the ions emitted from the astrocytes. Thus, astrocyte functions relate closely to brain functioning. Therefore, on the basis of the astrocyte functions, we consider the modeling of astrocyte functions in an artificial neural network (ANN).

Recently, many kinds of ANN models have been proposed and widely applied in modern technology. A multilayer perceptron (MLP) is a feed forward ANN comprising layers of neurons. In an MLP, the neurons comprise the neuron layers, and a neuron in one layer connects only with neurons in other layers. Then, we can obtain the ideal input-output relation of an ANN by setting up weights of network connections. In general, a backpropagation (BP) algorithm is used for determining connection weights [10]. Using the BP algorithm, an MLP can be used for various tasks such as pattern recognition, machine learning, and data mining. However, MLPs involve two problems. First, an MLP is trapped in a local minimum because the BP algorithm uses the steepest decent method. Second, an MLP does not have connections between neurons in the same layer. Furthermore, the neurons in a layer do not have position dependency in an MLP. Simulated annealing (SA) [11] can be used to improve the BP algorithm [12], [13]. In SA, the network searches a wide range with the objective function, and the searching range decreases with time. In this way, the SA can find a global solution. Moreover, a modeling of glial functions in an ANN was reported by A.B. Porto-Pazos, A. Alvarellos, and others [14]–[16]. These studies showed that glia improved the performance of a feed forward neural network. That work drew attention to the relationship between glia and neurons. In our model, we focus on the relationship among glia. We consider that a glial network can provide position relationships to neurons.

In this study, we propose a new MLP with a pulse glial chain* (PGC) inspired by the functions of astrocytes. All

Manuscript received February 19, 2015.
Manuscript revised August 28, 2015.

[†]The authors are with Tokushima University, Tokushima-shi, 770-8506 Japan.
[††]The author is with Xi'an Jiaotong University, Shaanxi 710049, China.
a) E-mail: ikuta@ee.tokushima-u.ac.jp
b) E-mail: uwate@ee.tokushima-u.ac.jp
c) E-mail: nishio@ee.tokushima-u.ac.jp
d) E-mail: gayang@mail.xjtu.edu.cn

DOI: 10.1587/transfun.E99.A.742

*We reported the concept and the basic investigation of a random glial network in an IEICE letter [18]. In the previous study, a glial cell generated a uniform random noise, whose amplitude was changed by the output of the connected neuron.

glia are individually connected to the neurons in the hidden layer of the MLP and have an interactive effect in the glial network. A neuron output is greater than a glial excitation threshold, hence it can excite a glial cell. The excited glial cell generates a pulse, and the pulse influences the excitation threshold of the neuron and the states of the neighboring glia. The glial pulse is attenuated exponentially and induces a pulse chain. The neighboring glia are excited and generate pulses subsequent to those of the first excited glial cell, thereby propagating the pulse in the glial network. The generated pulse is transmitted to the connected neuron and influences the excitation threshold of this neuron. This pulse accelerates the convergence of the weight update of the connected neuron because the pulse retains the large value of the neuron. A neuron output greater than the glial excitation threshold almost converges to a local optimum value, which is therefore retained by the pulse. The neighboring neurons obtain energy from the pulses propagated by other glia. The influence of a pulse is independent of the states of the neighboring neurons. Therefore, the neighboring neurons deviate from the local optimum values of their neighboring neurons. We consider that the pulse glial chain (PGC) accelerates the learning of the entire MLP network. This study confirms that the MLP with PGC has better performance than the standard MLP in learning time series, solving the Proben1 [17], and solving the two-spiral problem (TSP).

2. Proposed Method

In this study, we propose a PGC inspired by the features of astrocytes. The glia are individually connected to neurons in the hidden layer and influence neighboring glia. We connect glia only to the neurons in the hidden layer, because the number of neurons in the input and output layers depends on the simulation tasks. In addition, we consider that having connections within the same layer is important for the MLP. The conventional MLP already has connections between different layers. For the reasons stated above, we connect glia to neurons only to compare the glial effects in several different tasks. The proposed MLP with PGC is shown in Fig. 1.

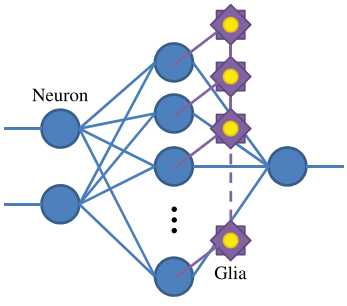

Fig. 1 MLP with PGC.

2.1 Glial Pulse Chain

All glia are connected to the nearest neurons in the hidden layer of an MLP and generate pulses according to the outputs of the connected neurons. Here, we define the glial output function as in Eq. (1).

$$\psi_i(t+1) = \begin{cases} 1, & \{\theta_n < y_i \cup \psi_{i+1}(t-D) = 1 \cup \\ & \quad \psi_{i-1}(t-D) = 1)\} \cap (t - \tau_i > \theta_g) \\ \gamma\psi_i(t), & else, \end{cases} \quad , \quad (1)$$

where ψ is the output of a glial cell, i is the position number of a glial cell in the hidden layer, y is the output of the connected neuron, θ_n is the excitation threshold of the glial cell, D is the delay time of a glial effect, τ is the time of the previous pulse generation, θ_g is the period of inactivity, and γ is the attenuation parameter. In the excitation condition of Eq. (1), $y_i > \theta_n$ indicates that the neuron excites the glial cell when the neuron output is greater than θ_n and has a constant value; thus, it is not changed with iteration. The glial cell is also excited by receiving a glial pulse from its neighboring glial cell, because the glial pulse requires time D to be transmitted to its neighboring glia, a condition described by $\psi_{i+1}(t-D) = 1$ and $\psi_{i-1}(t-D) = 1$. The glial cell has an inactivity period θ_g; hence, $t - \tau_i > \theta_g$ must be satisfied to generate a glial pulse. The glias do not learn based on the BP algorithm, even though the neurons are trained based on that algorithm. The generation pattern of glial pulses changes dynamically during MLP learning; an example pattern is shown in Fig. 2. In the figure, the first glial cell is excited and generates its corresponding pulse with the excitation condition of $y_i > \theta_n$ as shown in Eq. (1). The first glial cell receives neuron output greater than θ_n, and a pulse is propagated to the neighboring glial cells such that the transmitted pulse of the first glial cell excites the neighboring glial cells according to $\psi_{i+1}(t-D)$. Thus, the neighboring glial cells generate the pulse with a delay from the first glial cell. Furthermore, the tenth glial cell is excited at a time similar to that of the first glial cell, and the effect of the tenth glial cell is also propagated to the neighboring glia. Finally, the seventh glial cell is excited independently of the influences of the tenth and first glial cells. The seventh glial cell receives the pulse of the tenth glial cell; however, it is not excited by using these pulses, because the seventh had already begun the period of inactivity (i.e., to $t - \tau_i > \theta_g$.). The change in the generation pattern of pulses depends on the neurons' outputs, which are changed by BP learning, and the generation pattern Fig. 2(b) is also different from Fig. 2(a). From this example, we see that the generation pattern of pulses changes dynamically during learning.

Next, we focus on the responses of two glial cells. The outputs of the connected neurons and the responses of two neighboring glial cells are shown in Fig. 3. In this figure, glial cells one and two are connected to neurons one and two, respectively. Glial cell one neighbors glial cell two; thus, they influence each other. During the data iterations, the input learning data are switched in sequence with the

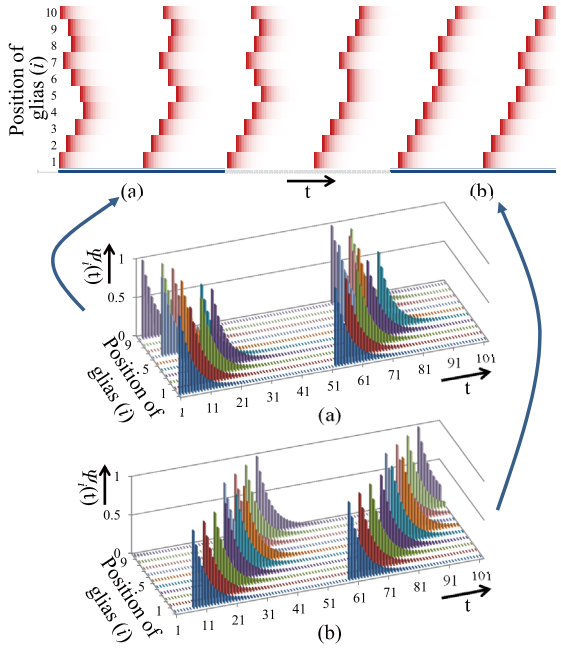

Fig. 2 An example of glial pulses ($D = 5$).

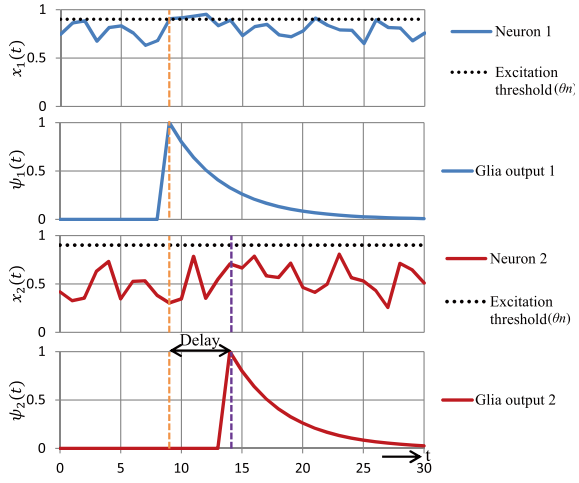

Fig. 3 Different patterns of the two glia excitations ($\theta_n = 0.9$, $D = 5$, $\gamma = 0.8$, and $\alpha = 0.5$).

iterations; thus, the output of a neuron changes dynamically with the iterations. The output of neuron one is greater than the excitation threshold of the connected glial cell one, and meanwhile, glial cell one is also excited by the output of neuron one. Actually, we just observe the pulse generation in glial cell one, because the output of neuron two is less than the excitation threshold of the connected glial cell two. However, glial cell one influences the state of glial cell two, and glial cell two is excited by glial cell one. Thus, glial cell two generates the pulse with a delay.

2.2 Neuron Updating Rule

A neuron is multi-input and single-output, and we can change its output by tuning connection weights between neurons. The standard updating rule for a neuron is defined as in Eq. (2).

$$y_i(t+1) = f\left(\sum_{j=1}^{n} w_{ij}(t)x_j(t) - \theta_i(t)\right), \tag{2}$$

where y is the output of the neuron, w is the weight of the connection, x is the input of the neuron, and θ is the excitation threshold of the neuron. In this equation, the weight of the connection and the excitation threshold of the neuron are learned based on the BP algorithm. Next, we show our proposed neuron updating rule. Because glia may increase the membrane potential of a neuron in a biological system, we add the glial effect ψ to the inner state of the neuron. (Glu, ATP, etc.) [19], [20], and these ions influence the membrane potential of the neuron. The inner state of the neuron increases as a result of the glial effect. In this study, this updating rule is used only for the neurons in the hidden layer. Since we emphasize the position relationship of neurons in the same layer, we arrange the glias in one dimension to enable observation of the position relationship of neurons. The updating rule proposed in this study is described by Eq. (3).

$$y_i(t+1) = f\left(\sum_{j=1}^{n} w_{ij}(t)x_j(t) - \theta_i(t) + \alpha\psi_i(t)\right), \tag{3}$$

where α is the weight of the glial effect. The peak of the generated pulse changes according to α. In this study, we choose an optimal value of α using heuristic search. In this expression, the weight of connection and the excitation threshold of the neuron are obtained based on the BP algorithm just as in the standard neuron updating rule. However, the glial effect is independent of learning. ψ is updated using Eq. (1). Equations (2) and (3) are used as sigmoidal functions to a neuron activating function expressed as in Eq. (4).

$$f(a) = \frac{1}{1+e^{-a}}, \tag{4}$$

where a is the inner state of the neuron. Several activating functions, such as tangent hyperbolic, using absolute value have been proposed for modeling MLP performance [21]. In this study, we use the sigmoidal function as the activating function. This function is basic and is used frequently [22] [23] because the derivative can easily be calculated, and it can easily be applied to the BP algorithm. In our method, the activation function does not have an essential role; therefore, we can change the sigmoidal function to other functions.

3. Simulations

In this section, we show our experimental results based on computer simulation of three different tasks. We use six

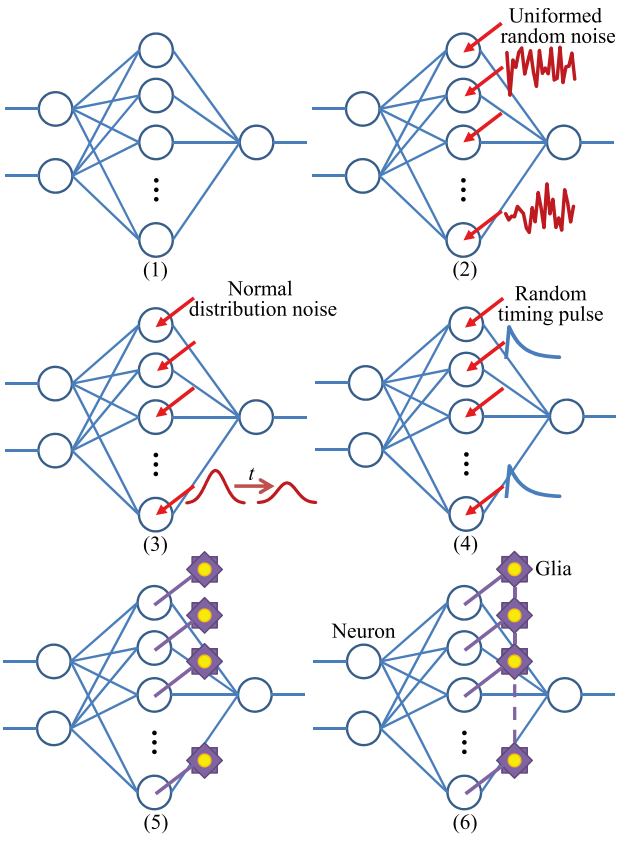

Fig. 4 Network structures of five MLPs.

types of MLP for the performance comparison.

(1) Conventional MLP
(2) MLP with random noise
(3) MLP with SA noise
(4) MLP with randomly timed pulses
(5) MLP with glial pulse
(6) MLP with PGC

The network structures for the six types of MLP models are shown in Fig. 4. A conventional MLP (1) does not have an external unit. An MLP with random noise (2) has a uniform random noise that influences the excitation threshold of neurons in the hidden layer. An MLP with SA noise (3) has a normally distributed noise that influences the excitation threshold of neurons in the hidden layer and whose amplitude decreases exponentially with iteration. An MLP with randomly timed pulses (4) has pulses at random times in the neurons in the hidden layer. In such an MLP, the neurons in the hidden layer have a pulse that influences the excitation threshold of the neuron, and this pulse is generated at random. In an MLP with a glial pulse (5), the glia respond to the output of the connected neuron in the same manner as the proposed MLP with PGC (6); however, the generated pulse is not propagated to neighboring glia. Thus, the generated pulse only increases the excitation threshold of the connected neuron. In the simulations, the optimal noise and pulse amplitude of each method is decided heuristically. In addition, we obtain the proper parameters of the PGC based on a large number of simulation experiments with various parameter values. Thus, the parameters are determined as $\theta_n = 0.9$, $D = 1$, $\theta_g = 45$, and $\gamma = 0.8$.

Here we use the mean square error (MSE) described by Eq. (5) as the error evaluation.

$$MSE = \frac{1}{N}\sum_{n=1}^{N}(T_n - O_n)^2, \tag{5}$$

where T is the target value, O is the output of MLP, and N is the number of learning data. In this study, we use the average, the minimum, the maximum, the standard deviation, and the accuracy of the MSE to evaluate the validation accuracy of the experimental result. MLP performance is better when the MSE is smaller and the validation accuracy is larger. We calculate the validation accuracy using a k-fold cross-validation estimate. Here, we fix the value of k to ten, and obtain the validation accuracy for each simulation, and we also apply a Wilcoxon signed-rank test to the experimental results. The Wilcoxon signed-rank test is one of the nonparametric tests. We compare two results obtained from different methods and obtain a sampling probability. If the sampling probability is less than 0.05, we conclude that the results have a broad distinction. In contrast, if the sampling probability is greater than 0.05, we are not sure whether two results are good or not.

3.1 Task 1: Learning Time Series

For the first task, we use successive chaotic time series as data sets, with the skew tent map for the generation of the time series described by Eq. (6).

$$\phi_i(t+1) = \begin{cases} \frac{1}{A_i}\phi_i(t) & (0 \le \phi_i(t) \le A_i) \\ \frac{1}{1-A_i}(1-\phi_i(t)) & (A_i < \phi_i(t) \le 1) \end{cases}, \tag{6}$$

We use $A_1 = 0.45$ and $A_2 = 0.55$. The generated chaotic time series vary with the value of A. The data set includes the two successive chaotic time series obtained with A_1 and A_2. An example of the successive chaotic time series is shown in Fig. 5. In this simulation, the MLP comprises three layers (connected 4-40-1), and the simulation conditions are as shown in Table 1. The column headings in the table indicate the following: the number of inputs is indicated for one round of learning, the number of classifications is for the Boolean classification of the input data, the training data sets are for the number of data sets used for learning, the unlearned data sets are for the number of data sets used to show the performance of unlearning data sets, and the validation is for the number of data sets used for validation obtained from ten-fold cross-validation.

We input four successive chaotic time series ($\phi_i(t)$, $\phi_i(t+1)$, $\phi_i(t+2)$ and $\phi_i(t+3)$) to the neurons in the input layer, and the MLP learns the correlating classification. And also, $\phi_1(t)$-$\phi_1(t+3)$ and $\phi_2(t)$-$\phi_2(t+3)$ are switched

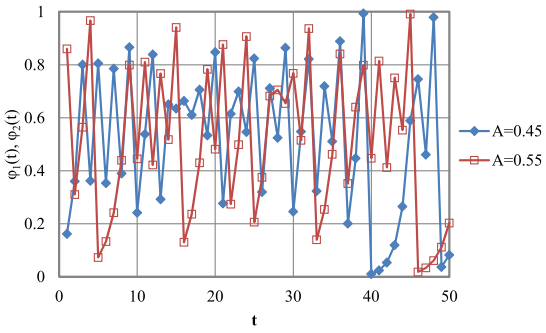

Fig. 5 Successive chaotic time series obtained by skew tent map.

Table 1 Conditions of chaotic time series.

Num. of inputs	Num. of class.	Training data sets	Unlearned data sets	Validation
4	2	200	200	20

Table 2 Learning performance of time series.

	Average	Minimum	Maximum	Std. Dev.	Accuracy
(1)	0.02868	0.00008	0.14008	0.02650	89.836
(2)	0.02464	0.00000	0.12016	0.02172	90.054
(3)	0.02841	0.00009	0.14011	0.02516	88.758
(4)	0.02804	0.00001	0.14003	0.02548	89.906
(5)	0.02840	0.00007	0.14010	0.02629	89.821
(6)	0.00756	0.00007	0.11087	0.01347	94.319

with time. We prepare ten data sets with each data set including 200 successive chaotic time series, obtained from ten different initial values. We use ten fold cross-validation for validation accuracy; thus, 180 and 20 data are used for learning and validation, respectively. We obtain the simulation results from 100 trials for each data set; thus, the total number of trials is 1,000, with each trial having 50,000 iterations.

Table 2 shows the statistical results from 1,000 trials. In this simulation, the validation accuracies are approximately 90 in the MLPs. From Table 2, we can see that the MLP with PGC (6) has the best average. For the maximum, the MLP with PGC (6) obtains the best value of 0.11087, and its standard deviation 0.01347, the smallest of all. However, for the minimum, the MLP with random noise (2) obtains the best value of 0.00000, though the other MLPs also obtain adequate results.

Table 3 shows the Wilcoxon signed-rank test for the MLPs. The sampling probability values are less than 0.05; thus, we can be sure of the differences in the average performance of the MLPs in Table 2.

Table 4 shows a classification performance of unlearning time series. We prepare the unlearning time series by using different initial values, with the number of data sets for unlearning time series being the same as for the learning data sets. We also provide the unlearning time series to the trained MLPs and compare the output of the MLPs with the true classification of the chaos generated using Eq. (6).

In this result, the validation accuracy becomes approximately 87 for every MLP. The proposed MLP with PGC (6) has the best average, maximum, and standard deviation. The MLP with random noise (2) has the best minimum. These results are similar to those for learning performance.

Table 3 Wilcoxon signed-rank test of learning performance of time series.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.000	0.009	0.000	0.000	0.000
(2)	0.000	-	0.000	0.011	0.000	0.000
(3)	0.009	0.000	-	0.000	0.006	0.000
(4)	0.000	0.011	0.000	-	0.000	0.000
(5)	0.000	0.000	0.006	0.000	-	0.000
(6)	0.000	0.000	0.000	0.000	0.000	-

Table 4 Classification performance of unlearning time series.

	Average	Minimum	Maximum	Std. Dev.	Accuracy
(1)	0.10164	0.00013	0.26408	0.06479	86.361
(2)	0.09946	0.00000	0.28116	0.06759	87.768
(3)	0.10164	0.00013	0.26408	0.06479	86.225
(4)	0.10094	0.00004	0.26800	0.06745	87.637
(5)	0.10179	0.00002	0.25382	0.06386	86.668
(6)	0.05681	0.00009	0.24771	0.05097	89.683

Table 5 Wilcoxon signed-rank test of classification performance of time series.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.258	1.000	0.417	0.959	0.000
(2)	0.000	-	0.258	0.724	0.154	0.000
(3)	0.000	0.000	-	0.417	0.959	0.000
(4)	0.000	0.011	0.000	-	0.386	0.000
(5)	0.000	0.000	0.006	0.000	-	0.000
(6)	0.000	0.000	0.000	0.000	0.000	-

Table 5 shows the Wilcoxon signed-rank test for the classification performance. The value of the sampling probability of the MLP with PGC (6) is less than 0.05 as compared with the others. We can be sure our method has better performance than the others, because average of the proposed MLP with PGC (6) in Table 4 is less than the others. Although the value of the sampling probability of the MLP with random noise (2) is smaller than the value of the conventional MLP, this value is larger than the others. Therefore, we cannot say that the MLP with random noise (2) is better than the others.

3.2 Task 2: Proben1

In this simulation, we use Proben1, the benchmark problems for ANNs [17]. We choose Cancer, Card, and Glass from the data sets of Proben1. Every data set has multivariable inputs and a Boolean supervised signal. The number of input dimensions and classifications of each task is shown in Table 6.

Here, the numbers of input dimensions and classifications for each task depends on the number of input and output neurons, respectively. In addition, the number of neurons in the hidden layer is 40, and the MLP comprises 9-40-2, 51-40-2 and 9-40-6 for solving Cancer, Card, and Glass, respectively. The data sets are as in Table 6. Thus, we obtain

Table 6 Data sets of Proben1.

Data	Num. of inputs	Num. of class.	Training data sets	Unlearn. data sets	Validation
Cancer	9	2	350	174	35
Card	51	2	345	172	34
Glass	9	6	107	53	10

Table 7 Learning performance of Proben1.

		Average	Minimum	Maximum	Std. Dev.	Accuracy
Cancer	(1)	0.00569	0.00286	0.01429	0.00132	93.891
	(2)	0.00592	0.00287	0.01429	0.00164	94.095
	(3)	0.00523	0.00001	0.00858	0.00129	94.404
	(4)	0.00578	0.00286	0.01429	0.00141	93.963
	(5)	0.00561	0.00286	0.02001	0.00207	93.983
	(6)	0.00492	0.00000	0.00858	0.00177	94.255
Card	(1)	0.01856	0.00581	0.03479	0.00653	83.553
	(2)	0.01886	0.00581	0.03479	0.00638	83.614
	(3)	0.01839	0.00581	0.03479	0.00657	89.195
	(4)	0.01864	0.00581	0.03479	0.00668	83.556
	(5)	0.01886	0.00294	0.03193	0.00610	83.606
	(6)	0.01769	0.00002	0.03189	0.00702	83.741
Glass	(1)	0.01058	0.00019	0.02942	0.00625	91.169
	(2)	0.01091	0.00177	0.02373	0.00547	91.452
	(3)	0.00952	0.00010	0.02245	0.00552	92.344
	(4)	0.01081	0.00022	0.02411	0.00584	91.476
	(5)	0.00903	0.00031	0.05939	0.00755	91.540
	(6)	0.00892	0.00031	0.02814	0.00628	91.750

the simulation results from 100 trials for each benchmark problem, and each trial has 50,000 iterations. Table 7 shows the learning performance of the MLPs for each benchmark problem.

In this simulation, the validation accuracies are approximately 94, 83, and 91, in Cancer, Card, and Glass, respectively. On an average, the proposed MLP with PGC (6) is the best of all for every benchmark task. In the minimum and maximum results, the proposed MLP with PGC (6) is the best for Cancer and Card, whereas the MLP with SA noise (3) has the best minimum and maximum results for Glass.

Tables 8–10 show results of the Wilcoxon signed-rank test for each model. In the learning of Cancer, the resulting evaluation of the proposed MLP with PGC (6) is less than 0.05. We can be sure the our method has better performance than the others, because average of the proposed MLP with PGC (6) in Table 7 is less than the others. In contrast, in the learning of Card, the experimental result of our method is greater than 0.05. Although we cannot verify the accuracy of the average, the average result of the proposed method is better than that of the others. In the evaluation of Glass, the value of the sampling probability of the MLP with PGC (6) is less than 0.05 as compared with the conventional MLP (1), the MLP with random noise (2), and the MLP with randomly timed pulses (4); however, these values are greater than 0.05 for the MLP with SA noise (3) and the MLP with a glial pulse (5).

Moreover, we show the classification performance of the unlearning data sets in Proben1. We input the unlearning data sets to the trained MLPs and compare outputs of

Table 8 Wilcoxon signed-rank test of learning performance of Cancer.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.000	0.490	0.000	0.000	0.000
(2)	0.000	-	0.000	0.000	0.000	0.000
(3)	0.490	0.000	-	0.062	0.000	0.004
(4)	0.000	0.000	0.062	-	0.000	0.000
(5)	0.000	0.000	0.000	0.000	-	0.000
(6)	0.000	0.000	0.004	0.000	0.000	-

Table 9 Wilcoxon signed-rank test of learning performance of Card.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.026	0.086	0.981	0.022	0.795
(2)	0.026	-	0.008	0.075	0.562	0.731
(3)	0.086	0.008	-	0.073	0.006	0.999
(4)	0.981	0.075	0.073	-	0.062	0.631
(5)	0.022	0.562	0.006	0.062	-	0.255
(6)	0.795	0.731	0.999	0.631	0.255	-

Table 10 Wilcoxon signed-rank test of learning performance of Glass.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.199	0.274	0.102	0.003	0.028
(2)	0.199	-	0.056	0.630	0.000	0.000
(3)	0.274	0.056	-	0.106	0.130	0.211
(4)	0.102	0.630	0.106	-	0.000	0.002
(5)	0.003	0.000	0.130	0.000	-	0.655
(6)	0.028	0.000	0.210	0.002	0.655	-

Table 11 Classification performance of unlearning data set of Proben1.

		Average	Minimum	Maximum	Std. Dev.	Accuracy
Cancer	(1)	0.01686	0.01275	0.01896	0.00110	98.195
	(2)	0.01662	0.01312	0.01875	0.00119	98.211
	(3)	0.01757	0.01325	0.02202	0.00094	98.220
	(4)	0.01671	0.01268	0.01904	0.00111	98.210
	(5)	0.01679	0.01358	0.02023	0.00118	98.207
	(6)	0.01501	0.01193	0.01872	0.00130	98.280
Card	(1)	0.08501	0.07414	0.10364	0.00599	91.431
	(2)	0.08498	0.07453	0.10399	0.00592	91.441
	(3)	0.08502	0.07411	0.10342	0.00604	91.437
	(4)	0.08499	0.07418	0.10365	0.00599	91.433
	(5)	0.08500	0.07395	0.10367	0.00593	91.435
	(6)	0.08284	0.07044	0.10400	0.00595	91.457
Glass	(1)	0.08837	0.08151	0.10872	0.00401	88.255
	(2)	0.08889	0.08235	0.10983	0.00430	90.996
	(3)	0.08609	0.08284	0.10279	0.00322	91.212
	(4)	0.08817	0.08161	0.10847	0.00394	91.074
	(5)	0.08709	0.08205	0.10814	0.00405	91.049
	(6)	0.08342	0.08073	0.10770	0.00409	91.033

the MLPs with the ideal classifications. In this simulation, the validation accuracies are approximately 98, 91, and 91, in Cancer, Card, and Glass, respectively. The classification performance is shown in Table 11, and the trend of the results is similar to that of Table 7. In every simulation, the MLP with PGC (6) obtains the best performance on average and the minimum for Cancer, Card, and Glass. The MLP with PGC (6) obtains the best maximum result only for Cancer; however, the MLP with SA noise (3) obtains the best maximum result for Card and Glass.

Tables 12–14 show the results of the Wilcoxon singed-rank test in the classification of Cancer, Card, and Glass, respectively. The value of the sampling probability of the

Table 12 Wilcoxon signed-rank test of classification performance of Cancer.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.001	0.000	0.000	0.001	0.000
(2)	0.001	-	0.000	0.149	0.034	0.000
(3)	0.000	0.000	-	0.000	0.000	0.000
(4)	0.000	0.149	0.000	-	0.209	0.000
(5)	0.001	0.034	0.000	0.209	-	0.000
(6)	0.000	0.000	0.000	0.000	0.000	-

Table 13 Wilcoxon signed-rank test of classification performance of Card.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.436	0.667	0.009	0.145	0.004
(2)	0.436	-	0.293	0.634	0.511	0.008
(3)	0.667	0.293	-	0.561	0.321	0.017
(4)	0.009	0.634	0.561	-	0.370	0.010
(5)	0.145	0.511	0.321	0.370	-	0.026
(6)	0.004	0.008	0.017	0.010	0.026	-

Table 14 Wilcoxon signed-rank test of classification performance of Glass.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.000	0.057	0.000	0.045	0.065
(2)	0.000	-	0.001	0.000	0.000	0.021
(3)	0.057	0.001	-	0.360	0.411	0.036
(4)	0.000	0.000	0.360	-	0.862	0.014
(5)	0.045	0.411	0.411	0.862	-	0.063
(6)	0.065	0.036	0.036	0.014	0.063	-

MLP with PGC (6) is less than 0.05 compared with the others in Cancer and Card. We can be sure that of the MLP with PGC (6) has better classification performance than the others. For Glass, the value of the sampling probability of the MLP with PGC(6) is less than 0.05 compared with the MLP with random noise (2), the MLP with SA noise (3), and the MLP with randomly timed pulses (4); however, the value of the sampling probability of this MLP is greater than 0.05 compared with the conventional MLP (1) and the MLP with a glial pulse (5).

3.3 Task 3: Two-Spiral Problem

For the next simulation, we use the two-spiral problem (TSP), a well-known highly nonlinear task for ANNs [24], [25]. This task has two sets of different spiral points. For learning, we input the coordinates of the spirals to the neurons in the input layer, and the MLPs learn the classification of two spiral points. Here, we use two different spirals comprising 98 and 130 points, respectively, as shown in Fig. 6. In the classification performance, we input coordinates between zero and one after learning. We obtain the output of the network, and determine which coordinates fits into which spirals. The simulation conditions of each spiral are as in Table 15. Figure 7 shows the ideal results of the classification of coordinates. We change the coordinates from zero to one in increments of 0.01 and input the coordinates to the trained MLP. Thus, the number of generated test data for the analyses of the classification performance is

Table 15 Conditions of TSP.

Data	Num. of inputs	Num. of class.	Training data sets	Unlearn. data sets	Validation
98	2	2	98	98	9
130	2	2	130	130	13

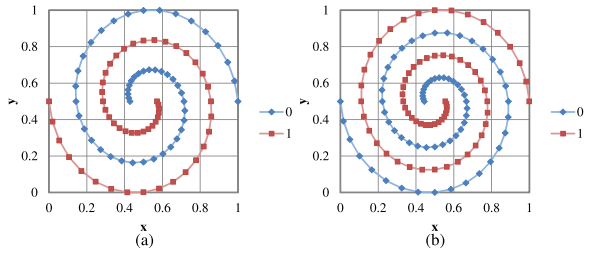

Fig. 6 Supervised points. (a) 98 spiral points. (b) 130 spiral points.

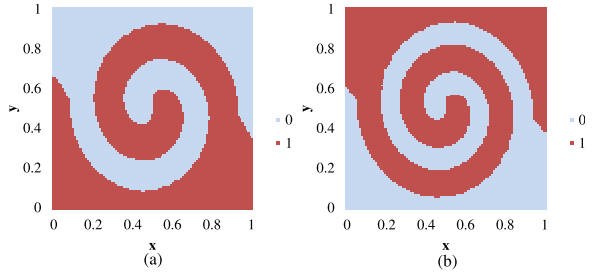

Fig. 7 Ideal classification results of two spirals. (a) 98 spiral points. (b) 130 spiral points.

101×101. Moreover, we ensure the ideal result by calculating a norm between coordinates and spiral points. Note that in this simulation, the MLP comprises 2-40-1 neurons.

3.3.1 Spirals Consisting of 98 Points

First, we show the experimental results from learning 98 points in Table 16. In this simulation, the validation accuracies are approximately 67; however, the validation accuracies of the MLPs are more decentralized than in the previous simulations. From Table 16, the performance of the conventional MLP (1) is the worst among all for the average error, because the conventional MLP (1) is often trapped in a local minimum. In the case of the MLP with PGC (6), the average error is the smallest of all. Energy is provided to the MLPs from several sources, and the noise providing energy to the MLP can generally escape from the local minimum. In addition, we can see the difference in the performance of the MLPs from this table.

Table 17 shows the evaluation of the Wilcoxon signed-rank test in the learning of 98 spiral points. The evaluation of the MLP with PGC (6) is less than 0.05 compared with the others; thus, we can be sure of the result of the MLP with PGC (6) for solving this task. The MLP with SA noise (3) has a better performance in Table 16; however, the evaluation of the Wilcoxon signed-rank test is greater than 0.05. Hence, we cannot say from this simulation that the MLP

Table 16 Learning performance for 98 points.

	Average	Minimum	Maximum	Std. Dev.	Accuracy
(1)	0.04153	0.00017	0.18387	0.02637	73.573
(2)	0.03711	0.00006	0.17352	0.02946	66.674
(3)	0.02957	0.00018	0.09213	0.02080	67.513
(4)	0.03666	0.00015	0.08208	0.02195	66.512
(5)	0.03249	0.00019	0.16390	0.02147	66.948
(6)	0.02072	0.00011	0.08192	0.01782	69.345

Table 17 Wilcoxon signed-rank test of learning performance of 98 points.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.085	0.001	0.008	0.008	0.000
(2)	0.085	-	0.173	0.592	0.592	0.000
(3)	0.001	0.173	-	0.211	0.211	0.002
(4)	0.446	0.375	0.010	-	0.225	0.000
(5)	0.008	0.592	0.211	0.225	-	0.000
(6)	0.000	0.000	0.002	0.000	0.000	-

Table 18 Classification performance of 98 spiral points.

	Average	Minimum	Maximum	Std. Dev.	Accuracy
(1)	0.15029	0.08085	0.21127	0.02434	76.811
(2)	0.13966	0.08083	0.20378	0.02879	77.574
(3)	0.13664	0.07611	0.21963	0.02837	81.971
(4)	0.14702	0.07965	0.20083	0.02553	77.171
(5)	0.13805	0.07529	0.20362	0.02468	78.740
(6)	0.12233	0.08140	0.17042	0.01939	80.434

Table 19 Wilcoxon signed-rank test of classification performance of 98 points.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.005	0.001	0.329	0.001	0.000
(2)	0.005	-	0.370	0.084	0.710	0.000
(3)	0.001	0.370	-	0.006	0.747	0.000
(4)	0.329	0.084	0.006	-	0.016	0.000
(5)	0.001	0.710	0.747	0.016	-	0.000
(6)	0.000	0.000	0.000	0.000	0.000	-

with SA noise (3) has better performance.

We show the classification results in Table 18. The results show a trend similar to that in Table 16. In general, the MLP has an overlearning problem when the MLP learns more than required, losing its generalization capability. However, the proposed MLP with PGC (6) can still obtain better approximation and classification performances.

Table 19 shows the evaluation of the Wilcoxon signed-rank test as the classification results of the 98 spiral points, and the trend of the evaluation is similar to that shown in Table 17. In addition, the evaluation of the MLP with PGC (6) ensures classification performance.

Figure 8 shows examples of the classification results. We can see from Fig. 8 that the MLP with PGC (6) draws only the two spirals; in contrast, the others are decoupled in some parts.

3.3.2 Spirals Consisting of 130 Points

Here, we show the results of the MLP learning 130 spiral points. For the TSP, simulation difficulty increases with increasing number of points. Table 20 shows the approximation results. In this simulation, the validation accuracy becomes approximately 60. The validation accuracies of the MLPs decrease from the results in Table 16. From this table, the differences of the performances are greater than for Table 16. The conventional MLP (1) falls into the local minimum more often than in the learning of 98 spiral points. Moreover, the MLP with random noise (2), the MLP with SA noise (3), the MLP with randomly timed pulses (4), and the MLP with glial pulse (5) have performance similar to the conventional MLP (1). We often hope that noise is efficient for highly nonlinear problems; however, we observe little improvement in learning performance by the methods of (2)–(5). The proposed MLP with PGC (6) also obtains energy from the glias; however, this MLP has a performance twice as good as the others for the average, the maximum, and the standard deviation. For the minimum, the MLP with random noise (2) obtains the best results. The minimum of the MLP with the PGC (6) is almost the same as that of the MLP with random noise (2). From these results, we can confirm that the proposed PGC is efficient improving MLP performance.

Table 21 shows the evaluation of the Wilcoxon signed-

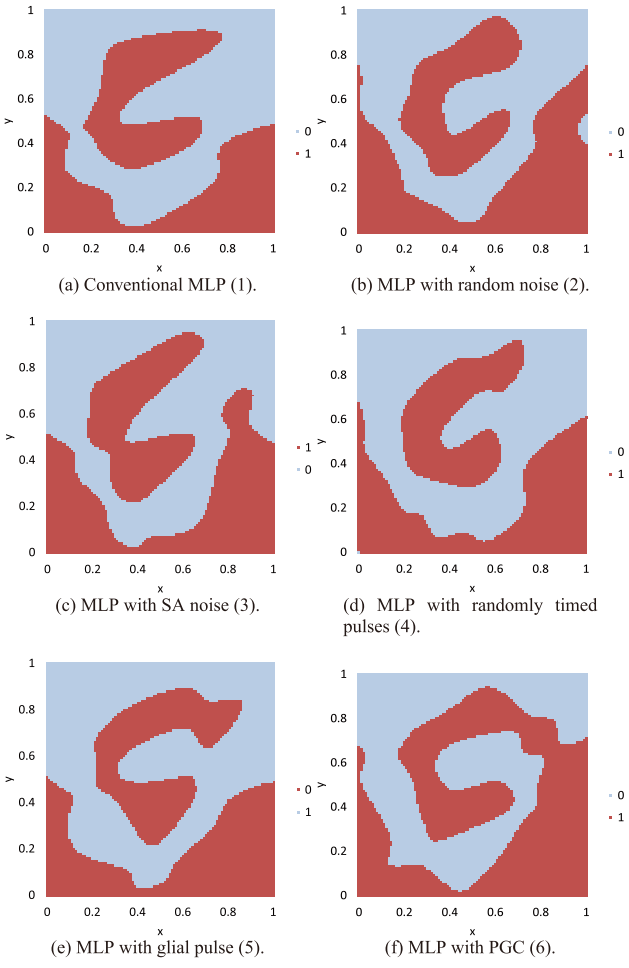

Fig. 8 Classification results of unlearned coordinates.

Table 20 Learning performance of 130 points.

	Average	Minimum	Maximum	Std. Dev.	Accuracy
(1)	0.12269	0.00831	0.23857	0.05554	61.194
(2)	0.10847	0.00047	0.24278	0.05742	62.847
(3)	0.09735	0.00107	0.24355	0.05356	64.960
(4)	0.11439	0.00740	0.26349	0.05742	59.386
(5)	0.09393	0.00130	0.25378	0.05544	59.386
(6)	0.03830	0.00063	0.12190	0.02589	62.368

Table 21 Wilcoxon signed-rank test of learning performance of 130 points.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.091	0.001	0.541	0.001	0.000
(2)	0.091	-	0.173	0.514	0.106	0.000
(3)	0.001	0.173	-	0.071	0.682	0.000
(4)	0.541	0.514	0.071	-	0.012	0.000
(5)	0.001	0.106	0.682	0.012	-	0.000
(6)	0.000	0.000	0.000	0.000	0.000	-

Table 22 Classification performance of 130 points.

	Average	Minimum	Maximum	Std. Dev.	Accuracy
(1)	0.21782	0.10565	0.29477	0.03858	71.968
(2)	0.19278	0.10460	0.33065	0.04434	72.701
(3)	0.19671	0.13272	0.28846	0.03166	75.660
(4)	0.20432	0.12082	0.31958	0.03851	72.451
(5)	0.19397	0.12303	0.29973	0.03730	74.182
(6)	0.14731	0.08792	0.23723	0.02826	75.870

Table 23 Wilcoxon signed-rank test of classification performance of 130 points.

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	-	0.091	0.000	0.017	0.000	0.000
(2)	0.000	-	0.403	0.057	0.667	0.000
(3)	0.000	0.403	-	0.186	0.543	0.000
(4)	0.017	0.057	0.186	-	0.051	0.000
(5)	0.000	0.667	0.543	0.051	-	0.000
(6)	0.000	0.000	0.000	0.000	0.000	-

rank test for the learning performance of 130 spiral points. The evaluation of the MLP with PGC (6) becomes zero; thus, we can be sure of the result in Table 20.

Table 22 shows the classification results for learning 130 spirals points, and the MLP with PGC (6) is the best of all by nearly every measure; therefore, we conclude that pulse propagation is important for MLP performance.

Table 23 shows the evaluation of the Wilcoxon signed-rank test of classification performance of 130 spiral points. The evaluation of the MLP with PGC (6) is less than 0.05 compared to the others; thus we can be sure of the differences in the average results in Table 22.

Figure 9 shows dependencies of learning and classification performances for the weight of glial effect α in the proposed MLP with PGC (6). In addition, we show a change in the performances of the MLP with random noise (2) and in the MLP with SA noise by changing the amplitudes of the uniform random and a SA noises, respectively. Note that we change the weight of the glial effect α and the amplitude of the uniform random noise from zero to one, and when α is equal to zero, the proposed MLP with PGC (6) is the

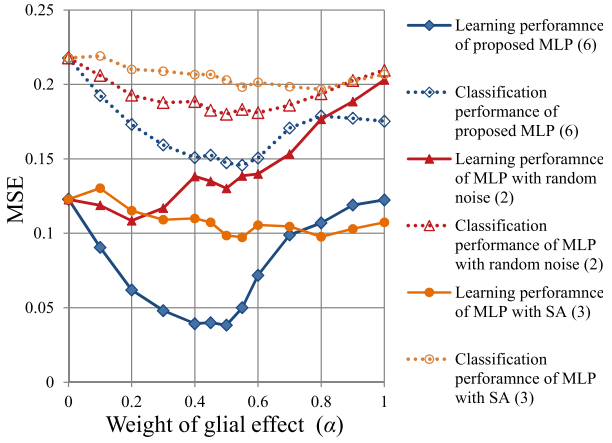

Fig. 9 Dependency of the learning and classification performances for the weight of glial effect α.

same as the conventional MLP (1). Generally, the learning performance corresponds with changes in the classification performance in the proposed MLP with PGC (6). The proposed MLP with PGC (6) has the best result for α equal to 0.5. In contrast, the MLP with random noise (2) has the best result with the amplitude of the uniform random noise equal to 0.2 in learning performance. In the case of the MLP with SA noise (3), the results are nearly unchanged with the amplitude of the noise at any time. We consider that the normal distribution noise rarely generates a large value in the transient state; thus, the dependency of the amplitude of the noise is lower with SA. From the difference in the results between the proposed MLP with PGC (6) and the MLP with random noise (2), the proposed MLP with PGC (6) can with a larger α than the MLP with random noise (2), because the glia provide energy to the network instantaneously through the generated pulse. Furthermore, we can use the large noise amplitude for the MLP with SA noise (3); however, the normal distribution noise is minimally effective for MLP performance. Thus, the MLP can reduce the error even if it receives a pulse of large amplitude from the glia. From this result, we conclude that the proposed PGC is more suitable for difficult tasks. In fact, our model improves MLP performance more in more difficult tasks.

Next, we show the dependency of learning performance on the number of neurons in the hidden layer of the MLPs in Fig. 10. We know that the performance of each MLP improves with the number of neurons, but once the number of neurons in the hidden layer is greater than 50, the learning performance decreases or remains on the same level, because the MLP requires long iterations for convergence when it contains many neurons. In the case of the MLP with PGC (6), the learning performance is similar to that of the others when the number of neurons is small. In addition, when the number of neurons increases, the difference in performance between the MLP with PGC (6) and the others increases. Thus, we can consider that the effect of the glia increases with the number of neurons. In the PGC, the

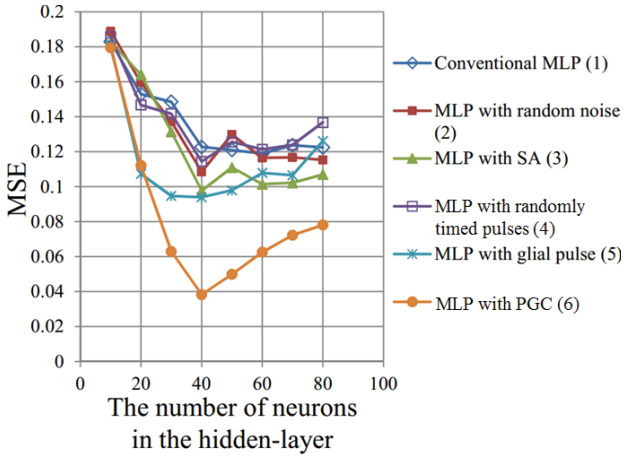

Fig. 10 Dependency of learning and classification performance for the number of neurons in the hidden layer.

pulse is propagated to the other glia. If the number of neurons is small, the pulse propagation finishes instantaneously; therefore, PGC has no effect on the MLP.

Finally, we show examples of the classification of unlearned coordinates results in Fig. 11. The conventional MLP (1) cannot represent the two spirals, and in the MLP with random noise (2) and the MLP with randomly timed pulses (4), the spirals are divided into several parts. The spirals of the MLP with SA noise (3) have a false area in the center of the image. The MLP with glial pulse (5) draw a part of spirals; however, there exists many errors in the upper area of the figure. The spirals of the MLP with PGC (6) are divided in only one part.

4. Discussions

In this section, we discuss the effects of the PGC on the MLP.

Firstly, we discuss the updating rule of the weights based on the BP algorithm. The updating rule of the weights between the hidden and output layers is as follows.

$$\Delta w_{kj} = \eta(T_k - O_k)O_k(1 - O_k)H_j, \tag{7}$$

where T is the target point, O is the output of the neuron in the output layer, H is the output of the neuron, and η is the learning coefficient in the hidden layer. The updating rule of the weights between the input and hidden layers is expressed by Eq. (8).

$$\Delta w_{ji} = \eta X_i H_j(1 - H_j)\sum_{k=1}^{n} w_{kj}(T_k - O_k)O_k(1 - O_k), \tag{8}$$

where X is the output of the neuron in the input layer. Equations (7) and (8) are proportional to H and $H(1 - H)$, respectively. The glial excitation depends on the outputs of neurons in the hidden layer; hence, we consider the relationships between the two equations and a neuron's output in the hidden layer. The weight update between the hidden and output layers, and the weight update between the input

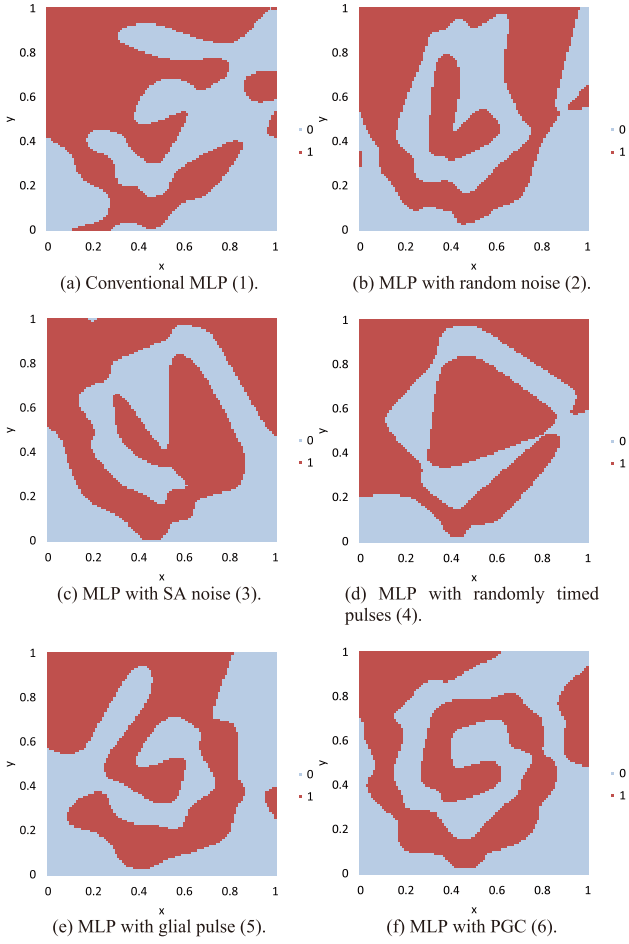

Fig. 11 Classification results of unlearned coordinates.

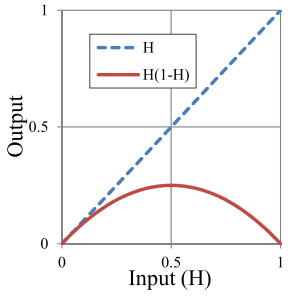

Fig. 12 Input-output characteristics of $H(1 - H)$.

and hidden layers depend on H and $H(1 - H)$, respectively. The descriptions of H and $H(1-H)$ are illustrated in Fig. 12. We can see from this figure that the weight update between the hidden and output layers increases with H and the weight update between the input and hidden layers decreases when H becomes greater than 0.5.

Next, we show total updates of the weights and the ratio of the number of pulse generations in each glial cell to the total iteration in Figs. 13 and 14. We obtain the ratios of number of iterations and the number of neurons whose output is greater than the excitation threshold of the glia to

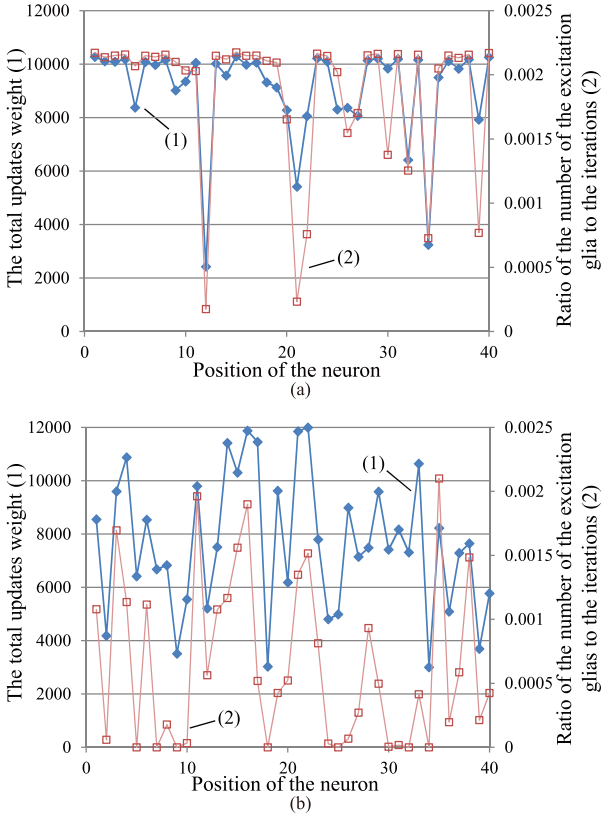

Fig. 13 Comparison of the total updated weights between the hidden layer and the output layer and the generated pulse. (a) The conventional MLP. (b) The proposed MLP.

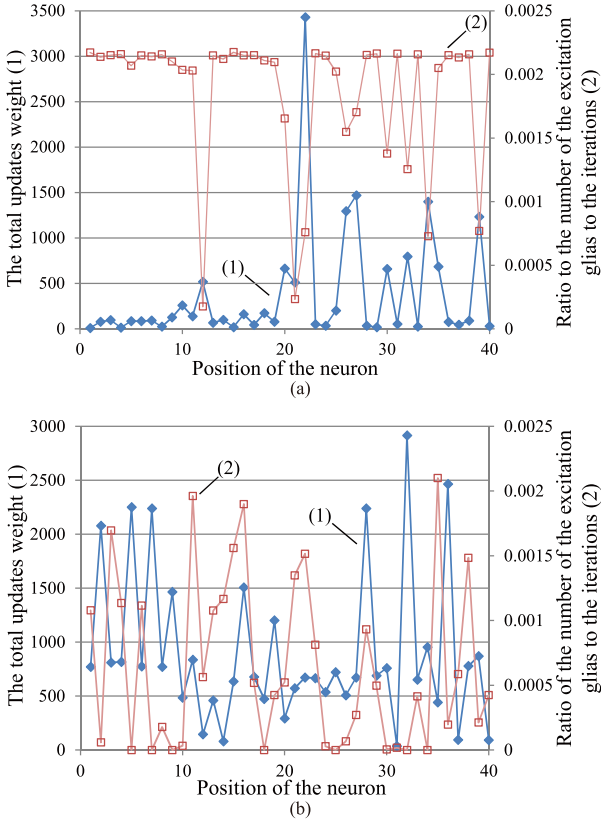

Fig. 14 Comparison of the total updated weights between the input layer and the hidden layer and the generated pulse. (a) The conventional MLP. (b) The proposed MLP.

the total number of iterations.

Figure 13 compares the characteristics of the hidden and output layers in the conventional and the proposed MLP. We can see that there is a positive correlation between the total updates of weights and the number of pulse generations in Figs. 13 (a) and (b). The characteristics relating to the hidden and output layers are in accordance with Eq. (7); however, the correlation between the total updates of weights and the number of pulse generations of the proposed MLP is much weaker than that of the conventional MLP. The correlation coefficient of the conventional MLP is 0.84, while the correlation coefficient of the proposed MLP is 0.65.

Figure 14 compares the characteristics of the input and hidden layers in the conventional and the proposed MLP. We can see that there is a negative correlation between the total updating weights and the number pulse generations in Figs. 14(a) and (b). The characteristics relating the input and hidden layers are in line with Eq. (8); however, the correlation between the total updates of weights and the number of pulse generations of the proposed MLP is much weaker than that of the conventional MLP. The correlation coefficient of the conventional MLP is -0.65, whereas the correlation coefficient of the proposed MLP is -0.21.

In the characteristics of the proposed MLP, we can see the position dependency of the total updates of the weights at the 20th and 35th neurons. The generated pulse increases the connected neuron output, and this pulse is propagated to the neighboring glia. The propagated pulse increases the outputs of the neighboring neurons irrespective of the inner state of the neighboring neurons, thereby displacing the learning points of the neighboring neurons. Thus, the updates of the weights in the neighboring neurons are changed, and the correlation coefficient of the proposed MLP is decreased. Therefore, we conclude that the neurons influence the neighboring neurons through the glial pulse generation, thereby improving the learning performance of the MLP.

Finally, we investigate the updated weights in detail when the MLPs learn chaotic time series. The updated weights are placed into five classes according to the state of the glial pulse. We periodically obtain the average of the updated weights of the hidden and output layers that meet the following requirements for a fixed period and unify the updated weights obtained from the same requirements of the neurons. The five types of updated weights are as follows: (A) $dw_{gp}(\tau_{gp})$ is the updated weight when the pulse input to the excitation threshold of the neuron and the glia is excited by the connected neuron, (B) $dw_{gp}(\tau_{gp} - 1)$ is the updated weight when the pulse is generated, (C) $dw_{rp}(\tau_{rp})$ is the updated weight when the neuron receives another glial pulse, (D) $dw_{rp}(\tau_{rp} - 1)$ is the updated weight when another glial pulse was propagated previously, and (E) $dw_{np}(\tau_{np})$ is the

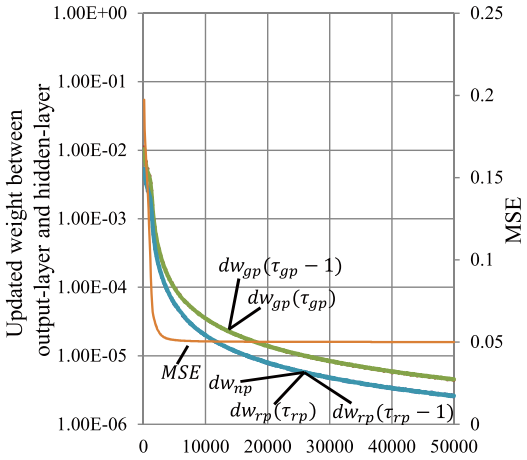

Fig. 15 Updated weights at pulse generation and received pulse in the conventional MLP (1).

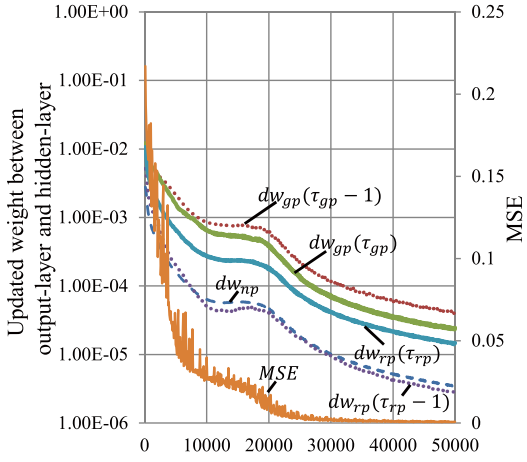

Fig. 16 Updated weights at pulse generation and received pulse in the MLP with PGC (6).

updated weight when the connected glial pulse becomes statistically small. Figures 15 and 16 show the updated weight in the conventional MLP (1) and the MLP with PGC (6), respectively, during the iteration.

In Fig. 15, we assume the pulse generation of glia based on the proposed method; however, this pulse is not input to the neuron. $dw_{gp}(\tau_{gp})$ overlaps with $dw_{gp}(\tau_{gp}-1)$, and $dw_{rp}(\tau_{rp})$ overlaps with $dw_{rp}(\tau_{rp}-1)$ and $dw_{np}(\tau_{np})$. According to this characteristic, the updated weight is not influenced for a short time. Moreover, the updated weight increases when the glia generate the pulse for the connected neuron because $dw_{gp}(\tau_{gp})$ and $dw_{gp}(\tau_{gp}-1)$ are larger than $dw_{rp}(\tau_{rp})$, $dw_{rp}(\tau_{rp}-1)$, and $dw_{np}(\tau_{np})$. In the error curve shown as MSE, the error reduction converges earlier.

In contrast, the error curve oscillates in Fig. 16, and every updated weight is different in comparison with Fig. 15. In particular, we can observe three characteristics, as follows.

1. $dw_{gp}(\tau_{gp})$ is smaller overall than $dw_{gp}(\tau_{gp}-1)$. This means that the updated weight when the glial cell generates a pulse from the connected neuron is smaller than that when the neuron receives a pulse from a connected glial cell. As a result of this characteristic, the glial pulse by which the glial cell is excited by the connected neuron decreases the updated weight, because the glial pulse increases the output of the connected neuron. The output of the connected neuron is already greater when the neuron receives the pulse of the connected glial cell; thus, the output of the connected neuron becomes closer to one as a result of the pulse. According to Fig. 12, the updated weight decreases when the neuron output is close to one.
2. $dw_{rp}(\tau_{rp})$ is larger overall than $dw_{rp}(\tau_{rp}-1)$. This means that the updated weight when the neuron receives the propagated pulse from another glial cell is greater than when another glial pulse was previously propagated. As a result of this characteristic, the other glial pulse increases the updated weight because this pulse increases the output neuron irrespective of the previous output of this neuron.
3. $dw_{np}(\tau_{np})$ is similarly small to $dw_{rp}(\tau_{rp}-1)$. As a result of this characteristic, the weight is slowly updated when the glial pulse becomes statistically small.

Overall, we conclude that the MLP with PGC (6) can find the various solutions because the updated weights are changed in various ways by the glial pulse. Moreover, the updated weight becomes small when the glial pulse becomes statistically small. Then, the MLP with PGC (6) can search specifically for the solution. In fact, the error curve shows that the MLP with PGC (6) obtains various solutions during the iteration and finds a better solution at the end of learning.

5. Conclusions

In this study, we proposed an MLP with PGC. The PGC was inspired by the biological features of glia, and we connected the glia to hidden layer neurons. Glial cells generate a pulse depending on the output of a connected neuron. The pulses affect the neighboring glial cells and the excitation threshold of the connected neuron. For updating weights, we also found that the position relationships depend on the generation of the pulses. Finally, we confirmed through three different simulations that the proposed MLP had better approximation and classification performance than the conventional and other MLPs. In our future works, we will verify the correlation between the position of the neuron in the hidden layer and the pulse generation.

Acknowledgment

This work was partly supported by MEXT/JSPS Grant-in-Aid for JSPS Fellows (24·10018), and by the Chinese national natural science foundation on under grants 61573272.

References

[1] P.G. Haydon, "Glia: Listening and talking to the synapse," Nat. Rev.

Neurosci., vol.2, no.3, pp.185–193, 2001.
[2] R.D. Fields and B. Stevens-Graham, "New insights into neuron-Glia communication," Science, vol.298, no.5593, pp.556–562, 2002.
[3] G.I Hatton and V. Parpura, Glia ⇔ neuronal signaling, Kluwer Academic Publishers, 2004.
[4] S. Koizumi, K. Fujishita, M. Tsuda, Y. Shigemoto-Mogami, and K. Inoue, "Dynamic inhibition of excitatory synaptic transmission by astrocyte-derived ATP in hippocampal cultures," Proc. National Academy of Sciences, vol.100, no.19, pp.11023–11028, 2011.
[5] S. Ozawa, "Role of glutamate transporters in excitatory synapses in cerebellar purkinje cells," Brain and Nerve, vol.59, pp.669–676, 2007.
[6] G. Perea and A. Araque, "Glial calcium signaling and neuron-Glia communication," Cell Calcium, vol.38, no.3-4, pp.375–382, 2005.
[7] S. Kriegler and S.Y. Chiu, "Calcium signaling of Glial cells along mammalian axons," J. Neurosci., vol.13, pp.4229–4245, 1993.
[8] M.P. Mattson and S.L. Chan, "Neuronal and Glial calcium signaling in Alzheimer's disease," Cell Calcium, vol.34, no.4-5, pp.385–397, 2003.
[9] N. Takata, T. Mishima, C. Hisatsune, T. Nagai, E. Ebisui, K. Mikoshiba, and H. Hirase, "Astrocyte calcium signaling transforms cholinergic modulation to cortical plasticity in vivo," J. Neurosci., vol.31, no.49, pp.18155–18165, 2011.
[10] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning representations by back-propagating errors," Nature, vol.323, no.6088, pp.533–536, 1986.
[11] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by simulated annealing," Science, vol.220, no.4598, pp.671–680, 1983.
[12] S. Amato, B. Apolloni, G. Caporali, U. Madesani, and A. Zanaboni, "Simulated annealing approach in backpropagation," Neurocomputing, vol.3, no.5-6, pp.207–220, 1991.
[13] C.B. Owen, A.M. Abunawass, and D.W. Ruck, "Application of simulated annealing to the backpropagation model improves convergence," Proc. SPIE, Science of Artificial Neural Networks II, pp.269–276, 1993.
[14] A. Porto, A. Pazos, and A. Araque, "Artificial neural networks based on brain circuits behaviour and genetic algorithms," Computational Intelligence and Bioinspired Systems, Lecture Notes in Computer Science, vol.3512, pp.99–106, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
[15] A.B. Porto-Pazos, N. Veiguela, P. Mesejo, M. Navarrete, A. Alvarellos, O. Ibanez, A. Pazos and A. Araque, "Artificial astrocytes improve neural network performance," PLos ONE, vol.6, no.4, e19109. doi: 10.1371/journal.pone.0019109, 2011.
[16] A. Alvarellos, A. Pazos, and A.B. Porto-Pazos, "Computational model of neuron-astrocyte interactions lead to improved efficacy in the performance of neural networks," Computational and Mathematical Methods in Medicine, vol.2012, Article ID 476324, 10 pages, 2012. doi: 10.1155/2012/476324.
[17] L. Prechelt, "PROBEN1 — A set of neural network benchmark problems and benchmarking rules," Technical report, Fakultat für Informatik, Universitat Karlsruhe, Germany, 1994.
[18] C. Ikuta, Y. Uwate, and Y. Nishio, "Multi-layer perceptron with Glial network for solving two-spiral problem," IEICE Trans. Fundamentals, vol.E94-A, no.9, pp.1864–1867, Sept. 2011.
[19] M. Morita, F. Yoshiki, and Y. Kudo, "Simultaneous imaging of phosphatidyl inositol metabolism and Ca^{2+} levels in PC12h cells," Biochem. Biophys. Res. Commun., vol.308, no.4, pp.673–678, Sept. 2003.
[20] A. Bal-Price, Z. Moneer, and G.C. Brown, "Nitric oxide induces rapid, calcium-dependent release of vesicular glutamate and ATP from cultured rat astrocytes," Glia, vol.40, no.3, pp.312–323, Oct. 2002.
[21] D.L. Elliott, "A better activation function for artificial neural networks," ISR Technical Report, TR 93-8, Jan. 1993.
[22] D.W. Ruck, S.K. Rogers, and M. Kabrisky, "Feature selection using a multilayer perceptron," J. Neural Netw. Comput., vol.2 no.2, pp.40–48, 1990.
[23] S.-I. Amari, H. Park, and K. Fukumizu, "Adaptive method of realizing natural gradient learning for multilayer perceptrons," Neural Comput., vol.12, no.6, pp.1399–1409, June 2000.
[24] J.R. Álvarez-Sánchez, "Injecting knowledge into the solution of the two-spiral problem," Neural Comput. Appl., vol.8, no.3, pp.265–272, 1999.
[25] H. Sasaki, T. Shiraishi, and S. Morishita, "High precision learning for neural networks by dynamic modification of their network structure," Dynamics & Design Conference, pp.411-1–411-6, 2004.

Chihiro Ikuta was born in Tokushima, Japan, in 1987. He received the B.E. and M.E. degrees in electrical and electronic engineering from Tokushima University, Tokushima, Japan, in 2010 and 2012, respectively. He was a Research Fellow (DC1) of the Japan Society for the Promotion Science (JSPS) from April 2012 to March 2015. He is currently working towards the Doctorate degree in the Department of Electrical Engineering, Tokushima University. He has been currently working at National Institute of Technology, Anan College as Assistant Professor from April 2015. His research interests include neural network and image processing. He is a student member of the IEEE.

Yoko Uwate (S'02-M'07) was born in Tokushima, Japan, in 1980. She received the B.E., M.E., and Ph.D. degrees in electrical and electronic engineering from Tokushima University, Tokushima, Japan, in 2003, 2005 and 2006, respectively. During October 2006–March 2008, she was a Postdoctral Resarch Fellow (PD) of the Japan Society for the Promotion Science (JSPS) at the same university, and she was also Visiting Post Doctoral Research Fellow, Institute of Neuroinformatics (INI), University and ETH Zurich. From April 2008 to March 2010, she worked as a Postdoctoral Fellow for Research Abroad of Japan Society for the Promotion Science (JSPS) at Institute of Neuroinformatics (INI), University and ETH Zurich. Since April 2010, she has been currently working at Tokushima University as Assistant Professor. Her research interests include complex phenomena in chaotic circuits and neural networks. She was an Associate Editor of the IEEE Transactions on Circuits and Systems–I: Regular Papers during Dec. 2012–Dec. 2013. She is currently a member of the IEEE CAS Society Board of Governors (from 2013). She is a member of the IEEE, the IEICE and the RISP.

Yoshifumi Nishio received the B.E., M.E., and Ph.D. degrees in electrical engineering from Keio University, Yokohama Japan, in 1988, 1990, and 1993, respectively. In 1993, he joined the Department of Electrical and Electronic Engineering at Tokushima University, Tokushima Japan, where he is currently a Professor. From May 2000 he spent a year in the Laboratory of Nonlinear Systems (LANOS) at the Swiss Federal Institute of Technology Lausanne (EPFL) as a Visiting Professor. His research interests are in the areas of nonlinear circuits engineering, including analysis and application of chaos in electrical circuits, analysis of synchronization in coupled oscillatory circuits, development of analyzing methods for nonlinear circuits, theory and application of cellular neural networks, neural network architecture, evolutionary computation, and biosignal processing. He was the Chair of the IEEE CAS Society Technical Committee on Nonlinear Circuits and Systems (NCAS) during 2004–2005 and the Steering Committee Secretary of the IEICE Research Society of Nonlinear Theory and its Applications (NOLTA) during 2004–2007, and is currently the Chair of the IEEE CAS Society Shikoku Chapter (from 2011) and a member of the IEEE CAS Society Board of Governors (from 2012). He was an Editor of the IEICE Fundamentals Review during 2007–2012 and an Associate Editor of the IEEE Transactions on Circuits and Systems–I: Regular Papers during 2004–2005, the IEEE CAS Magazine during 2008–2009, and the IEEE Transactions on Circuits and Systems–II: Express Briefs during 2012–2013. He is currently serving as a Secretary for the NOLTA, IEICE (from 2009) and an Associate Editor of the IEEE CAS Society Newsletter (from 2007). He is a senior member of the IEEE, and a member of the IEICE and the RISP.

Guaon Yang received the B.Eng. degree in automatic control engineering from Jilin University China, and M.Eng. degree from Tokyo Metropolitan University Japan in 1986 and 1993, respectively. He was a researcher fellow with Hertz co. Ltd in Tokyo Japan, from 1993 to 2001. Since May 2001, He joined the institute of artificial intelligence and robotics of Xi'an Jiaotong University China, and received the Ph.D. degree in the Department of Control Science and Engineering, in the school of electronic and information of Xi'an Jiaotong University China, in 2007. Since January 2011, he has been an associate professor of the department of automation science and technology in the school of electronic and information of Xi'an Jiaotong University. His main research interests include wavelets, image and multidimensional signal processing, multiscale geometric analysis, and sparse representation theory and applications.