DE GRUYTER
OPEN

JAISCR, 2018, Vol. 8, No. 1, pp. 5 – 29
10.1515/jaiscr-2018-0001

SELF-ASSIMILATION FOR SOLVING EXCESSIVE INFORMATION ACQUISITION IN POTENTIAL LEARNING

Ryotaro Kamimura[1] and Tsubasa Kitago[2]

[1]*IT Education Center, Tokai University*
4-1-1 Kitakaname, Hiratsuka, Kanagawa 259-1292, Japan
Email: ryo@keyaki.cc.u-tokai.ac.jp

[2]*Department of Politics and Economics, Tokai University*
4-1-1 Kitakaname, Hiratsuka, Kanagawa 259-1292, Japan

Submitted: 31st March 2017; accepted: 19th April 2017

Abstract

The present paper aims to propose a new computational method for potential learning to improve generalization and interpretation. Potential learning has been proposed to simplify the computational procedures of information maximization and to specify which neurons should be fired. However, it is often the case that potential learning sometimes absorbs too much information content on input patterns in the early stage of learning, which tends to degrade generalization performance. This can be solved by making potential learning as slow as possible. Accordingly, we here propose a procedure called "self-assimilation" in which connection weights are accentuated by their characteristics observed in the specific learning step. This makes it possible to predict future connection weights in the early stage of learning. Thus, it is possible to improve generalization by slow learning and at the same time to improve the interpretation of connection weights via the enhanced characteristics of the connection weights. The method was applied to an artificial data set, as well as a real data set of counter services at a local government office in the Tokyo metropolitan area. The results show that improved generalization was observed by making learning as slow as possible. In addition, the number of strong connection weights became smaller for better interpretation by self-assimilation.

Keywords: neural networks, learning, excessive information acquisition, self-assimilation method

1 Introduction

1.1 Problems of Information-Theoretic Methods

Information-theoretic methods have played important roles in neural information where information content on input patterns is maximized or minimized, depending on the specific problem, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. In these methods, there has been persistent difficulty in computing information content in terms of entropy or mutual information. Thus, there have been many attempts to reduce computational complexity by supposing that neurons are distributed independently or uniformly [11, 12, 13]. However, they have not necessarily been successful in simplifying the computational procedures. In addition, there is another problem, namely, the information-theoretic method cannot specify which neurons should be fired. For example, in a state of maximum information, one neuron fires while all others cease to do so. To maximize information we should choose which neuron is fired, but the criteria to do so remain uncertain. Thus, in order for the information-theoretic methods to be used practically, the problem of complex computation and uncertain specifications of fired neurons must be solved.

1.2 Potential Learning

Potential learning has been previously proposed to simplify computational procedures and to specify which neuron is fired in information maximization [14, 15, 16, 17]. When the information content of neurons is supposed to be maximized, then just one neuron should fire, while all the others cease to do so. In potential learning, this neuron should have maximum potentiality and be fired maximally. The potentiality of neurons represents the neurons' ability to respond appropriately to as many different situations as possible. For the first approximation, the potentiality has been defined as the variance of neurons.

Potential learning has three strong points, namely, simple computational procedures, specification of which neuron to fire, and independence of operations. By supposing the potentiality of neurons, it is possible to maximize mutual information simply by changing the potential parameter; no complex computational procedures are needed. Second, the method can explicitly determine which neuron is to be fired. Because potential learning aims to increase the potentiality of neurons, a neuron to be fired should be one with higher potentiality. Third, potentiality maximization or information maximization can be applied independently of error minimization. The potentiality is given in the initial stage of learning, and learning is considered to be a procedure to assimilate this potentiality into connection weights. Thus, there is no complex parameter control to compromise between potentiality maximization and error minimization as usefully done in the conventional methods. Thus, potential learning can simplify the computational procedures and at the same time extract the meaning of fired neurons.

1.3 Self-Assimilation for Excessive Information

Potential learning has been proved to be useful in improving the generalization and interpretation of neural networks. However, there is a problem called "excessive information acquisition", which occurs when potential learning tends to absorb too much information on input patterns even in the early stage of learning steps. In particular, when the problems become more complex and practical, this tendency became clearer. Since information maximization corresponds to a decrease in the number of neurons to be used in learning, it becomes easier to interpret the behaviors of neurons in a state with excessive information content. However, excessive information or excessive simplification has unfavorable effects on generalization performance. This means that the excessive information acquisition naturally degrades generalization performance. One solution for this problem is to make learning as slow as possible to prevent neural networks from acquiring too much information content. Because of this slow learning, potential learning requires a large number of learning steps to improve generalization performance,

To overcome this excessive information acquisition problem, we here propose a procedure called "self-assimilation", where the characteristics of connection weights are forced to be accentuated by assimilating the characteristics of the weights themselves. Because the characteristics of connection weights become gradually clearer when the information content becomes higher, this accentuation of connection weights is able to predict the future characteristics of connection weights from their present state. The self-assimilation aims to extract the main features of connection weights in stages with low information content, where neurons do not necessarily respond to input patterns explicitly. In other words, we try to infer the main characteristic of connection weights even in the early stages of learning. By this method, it is possible that the learning can proceed as slowly as possible to improve generalization, while the main features can be detected in the early stage of learning.

1.4 Paper Organization

In Section 2, we briefly explain the concept of potentiality assimilation and self-assimilation. We then explain the computational procedures for potential learning and self-assimilation, and introduce self-assimilated connection weights and mutual information. In addition, we explain how to interpret connection weights obtained by different data sets and initial conditions, which is called "collective interpretation". We provide the computational procedures for assimilating potentiality. In Section 3, we present two experimental results on an artificial data set and on a real data set on the counter services of a local government office in the Tokyo metropolitan area. The experimental results show that generalization was improved by slow learning and smaller parameter values, and that self-assimilation led to the detection of clear characteristics of connection weights, even in the early stages of learning. For the real data set, the final results by our method suggest that the configuration of the counters is the main reason behind stopping the processes of the counter services.

2 Theory and Computational Methods

2.1 Potentiality Assimilation and Self-Assimilation

2.1.1 Potential Learning for Interpretation

Information maximization methods have long been used to interpret final representations by simplifying network configurations [18, 19, 20]. In this case, the information content can be stored in a small number of neurons and connection weights. As mentioned in the introduction section, the information maximization methods require heavy computation to compute the entropy and corresponding information content. To address this issue, we have previously introduced potential learning to simplify the computational procedures [14, 15, 16, 17]. In maximum information states, only one neuron fires while all others cease to do so. Thus, one of the possible ways to realize this situation is to optimally determine which neuron should be fired. Potential learning aims to do precisely this by identifying the neuron with maximum potentiality. In previous studies, potentiality has been defined in terms of variance. Thus, the neuron to be fired should have maximum variance. This potentiality is assimilated in connection weights as initial conditions as shown in Figure 1. As shown in Figure 1(a), in the first step of potential learning, a neural network is first trained to determine the estimated potentiality of neurons. In the second step in Figure 1(b), this potentiality is assimilated into connection weights. These steps continue along their predetermined course. As shown in Figure 1(c), in the final step, the potentiality of a hidden neuron becomes the largest and all the other neurons tend to be much smaller.

As shown in Figure 1, let w^t_{jk} denote connection weights from the kth input neuron to the jth hidden neuron for the tth data set. Then, the potentiality is

$$^{t}v_j = \frac{1}{L-1}\sum_{k=1}^{L}\left(^{t}w_{jk} - {}^{t}w_j\right)^2, \tag{1}$$

where L is the number of input neurons, and

$$^{t}w_j = \frac{1}{L}\sum_{k=1}^{L} {}^{t}w_{jk}. \tag{2}$$

The potentiality is normalized as

$$p(j|t) = \frac{^{t}v_j}{\sum_{m=1}^{M} {}^{t}v_m}, \tag{3}$$

where M is the number of hidden neurons. Then, the potential information is

$$PI = \sum_{t=1}^{T} p(t) \sum_{j=1}^{M} p(j|t) \log \frac{p(j|t)}{p(j)}, \tag{4}$$

where T is the number of input patterns and $p(j)$ denotes the average firing probability for the jth hidden neuron

$$p(j) = \frac{1}{T}\sum_{t=1}^{T} p(j|t). \tag{5}$$

The potential information can be increased by assimilating the relative potentiality

$$^{t}\phi^r_j = \left(\frac{^{t}v_j}{^{t}v_{\max}}\right)^r, \tag{6}$$

where $v_{\max}$ is the maximum potentiality and r denotes the potential parameter, having positive values. The first step uses the ordinary back-propagation with the early stopping as shown in Figure 1(a) to determine the potentiality of connection weights. Then, in the second step in Figure

1(b), initial weights are added to the potentiality computed in the first step. To obtain the weights of the $n+1$th step, we must add the potentiality computed in the nth step

$$ {}^{t}w_{jk}(n+1) = {}^{t}w_{jk}\, {}^{t}\phi_{j}^{r}(n). \tag{7}$$

Then, learning is performed until the early stopping criterion is met. By increasing the parameter r, the potential information can be increased gradually.

2.1.2 Self-Assimilation for Enhancement

Self-assimilation is a method to enhance the characteristics of connection weights themselves. Figure 2 shows the process of potential learning and self-assimilation. After first determining the potentiality of neurons, potentiality assimilation is applied by changing the potential parameter in Figure 2(a). Then, the self-assimilation is applied to enhance connection weights by changing the potential parameter in Figure 2(b). As already mentioned, potential information tends to increase gradually in learning. Self-assimilation tries to infer what the states of connection weights will be in the later stages of learning at an earlier point. Thus, self-assimilation aims to predict future characteristics by observing the characteristics of the present stage.

Self-assimilation implies that connection weights can be transformed, reflecting their characteristics of potentiality. The self-assimilated potentiality is

$$ {}^{t}v_{j}^{r} = \frac{1}{L-1}\sum_{k=1}^{L}\left({}^{t}w_{jk}^{r} - {}^{t}w_{j}^{r}\right)^{2}. \tag{8}$$

where

$$ {}^{t}w_{j}^{r} = \frac{1}{L}\sum_{k=1}^{L} {}^{t}w_{jk}^{r}. \tag{9}$$

The potentiality is normalized as

$$ p^{r}(j|t) = \frac{{}^{t}v_{j}^{r}}{\sum_{m=1}^{M} {}^{t}v_{m}^{r}}. \tag{10}$$

Then, we have self-assimilated mutual information

$$ PI^{r} = \sum_{t=1}^{T}\sum_{j=1}^{M} p(t)p^{r}(j|t)\log\frac{p^{r}(j|t)}{p^{r}(j)}. \tag{11}$$

Finally, we note difference between simple potential learning and self-assimilation. In simple potential learning, the potentiality is forced to be assimilated into connection weights by repeating the processes of assimilation. On the other hand, in self-assimilation, weights are not updated; rather, only the parameter r is changed to control the potentiality in Figure 2(b). First, the parameter r is chosen so as to improve generalization performance. After learning is complete, connection weights are transformed by the relative potentiality computed from the connection weights themselves. This transformational operation has the effect of predicting the future characteristics of connection weights. Thus, at the very early stages of learning, we can predict the final or future characteristics of connection weights.

2.1.3 Collective Interpretation

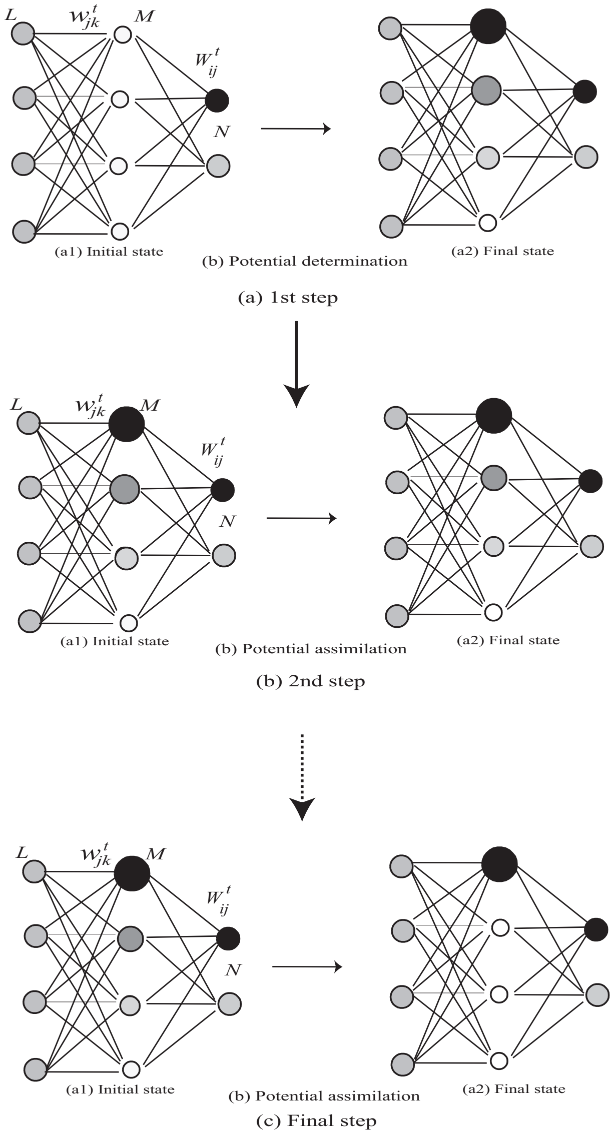
The neural networks tend to produce a variety of connection weights, depending on input patterns and initial conditions; we thus need to develop a method to interpret them all. Usually, the production of many different kinds of connection weights has been considered to be one of the main shortcomings of neural networks, compared with conventional statistical methods such as regression analysis. However, we consider the production of many weights to be one of the main strengths of neural networks, if those weights can be interpreted collectively.

This means that we try to produce many different kinds of connection weights in Figure 3. The main characteristics of these connection weights can be inferred from their average. In other words, we consider main characteristics to be those common to many different types of connection weights. Thus, we should develop a method to interpret the collective behaviors of connection weights: a new method of interpretation and a new way of visualize the final results from neural networks. The collective interpretation is robust to small changes in data sets and initial conditions. Thus, the method can stabilize the final interpretation of results, which has been a serious problem with neural networks.

Let ${}^{t}w_{jk}^{r}$ and ${}^{t}w_{ij}^{r}$ denote input-hidden and hidden-output connection weights, then the average weights called "collective weights" are computed by

$$ \bar{w}_{ik}^{r} = \frac{1}{TM}\sum_{t=1}^{T}\sum_{j=1}^{M} {}^{t}w_{jk}^{r}\,\mathrm{sign}({}^{t}W_{ij}^{r}), \tag{12}$$

where $\mathrm{sign}(W_{ij})$denotes the sign of hidden output



(a1) Initial state

(b) Potential determination

(a2) Final state

(a) 1st step

(a1) Initial state

(b) Potential assimilation

(a2) Final state

(b) 2nd step

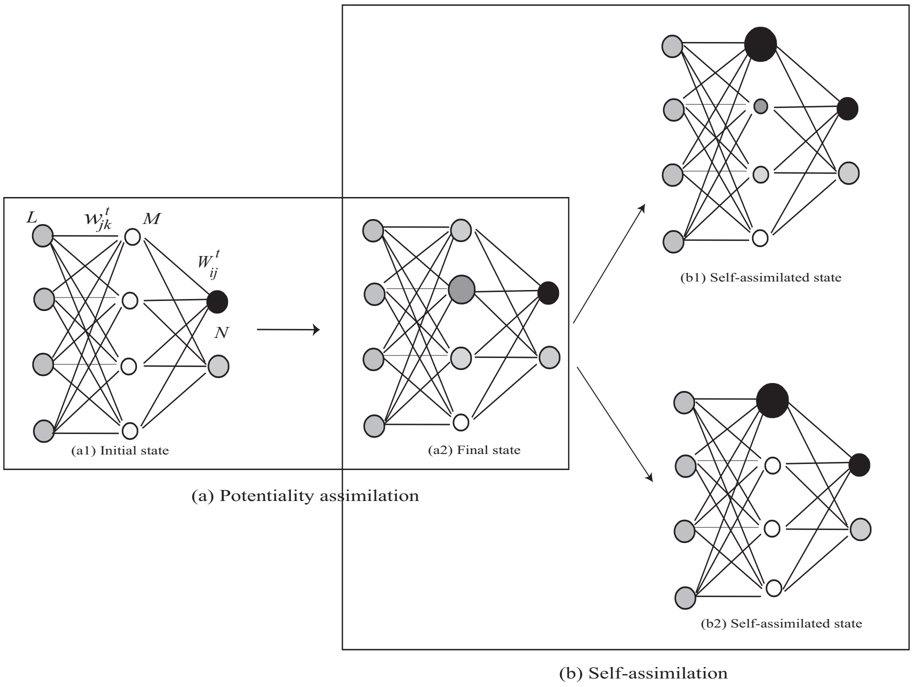
(a1) Initial state

(b) Potential assimilation

(a2) Final state

(c) Final step

Figure 1. Potential learning with potentiality determination and assimilation



(a) Potentiality assimilation

(b) Self-assimilation

Figure 2. Potentiality assimilation and self-assimilation

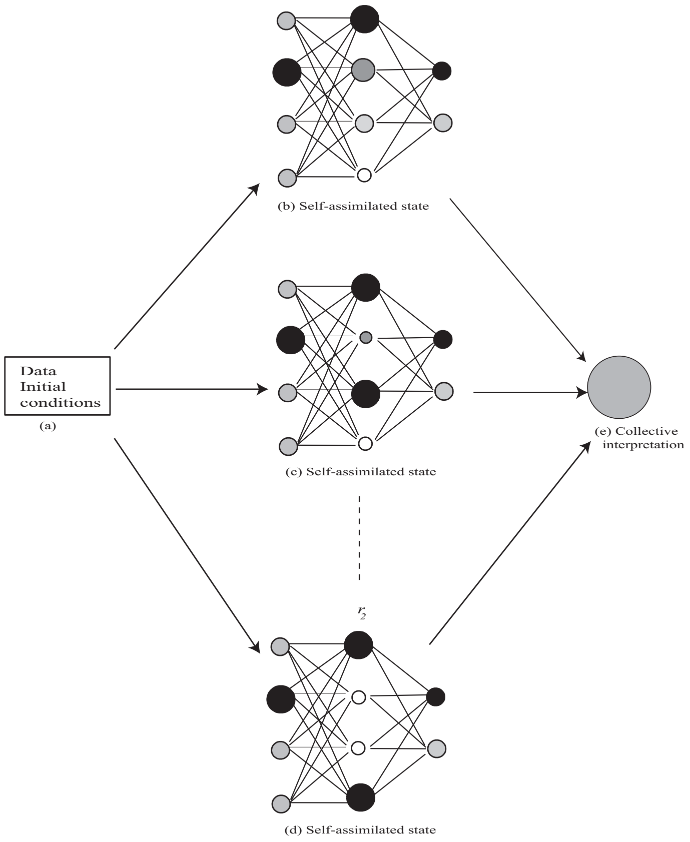

(b) Self-assimilated state
Data
Initial
conditions
(a)
(c) Self-assimilated state
(e) Collective
interpretation
r_2
(d) Self-assimilated state

Figure 3. Collective production and interpretation connection weights into the ith output neuron. By the collective weight, we try to see the main characteristics of connection weights by averaging all weights for different input patterns, weighted by the sings of hidden-output connection weights.

3 Results and Discussion

3.1 Artificial Data

3.1.1 Experimental Outline

The first experiment aims to show that the new method could improve generalization performance by detecting the importance of input neurons (variables). In the experiment, the artificial data set was composed of two sets of random values in Figure 4(a). The problem was to classify the data set into two classes. As shown in Figure 4(a), the variable x_1 was more important in the classification, and that methods should be able to detect this difference. The number of input patterns was 2,000 and the data set was divided into the training (70%), validation (15%) and testing (15%); ten different data sets were taken. Figure 4(b) shows the network architecture where two input, ten hidden and two output neurons were used. All the parameter values for learning were default ones in Matlab neural networks for easily reproducing the present results.

3.1.2 Potential Information Maximization

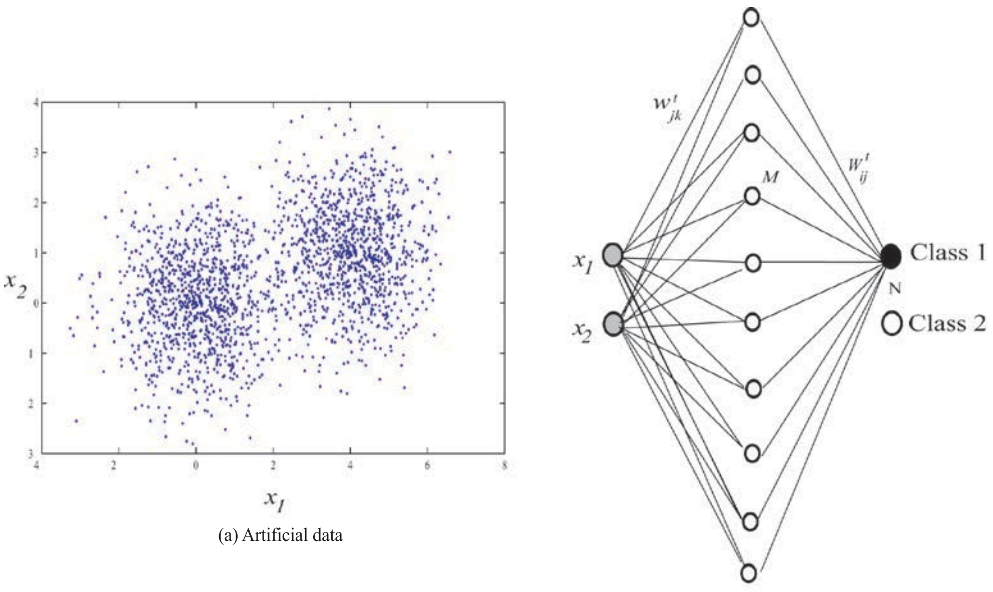
The potential information increased gradually and became close to 0.9 as the parameter increased. Figure 5 shows potential information and its corresponding generalization errors. As shown in Figure 5(a), when the parameter r increased, the potential information increased gradually. When the parameter r was 0.01, the potential information did not increase even when the number of steps increased. When the parameter r was increased to 0.1, the information increased slightly. When the parameter r was 1, the information became close to 0.9 with seven steps. When the parameter r was increased to ten, the information shifted between 0.8 and 0.9 and then increased slowly and surpassed 0.9. These results show that when the parameter r was increased from one to ten, the potential information was forced to increase rapidly.

As shown in Figure 5(b), generalization errors decreased when the number of steps increased. In particular, when the parameter r was 0.1, the lowest generalization errors were obtained. When the parameter r was 0.01, the generalization errors deceased very steadily, though the generalization errors were rather large. When the parameter was 0.1, the generalization errors decreased to the lowest point with seven steps. When the parameter r was further increased to one, the generalization errors deceased with seven steps, and then inversely increased. When the parameter r was 10, the generalization errors deceased greatly with two steps and then later fluctuated. These results show that too much information with higher parameter values prevented neural networks from improving generalization performance.

Figure 6 shows connection weights by four different parameter values for the artificial data set. When the parameter r was 0.01 in Figure 6(a), many strong positive and negative weights could be seen, even when the number of steps was increased from one in Figure6(a1) to ten in Figure 6(a5). When the parameter was 0.1, connection weights from the second input neuron became slightly weaker. When the parameter r increased to one in Figure 6(c) and ten in Figure 6(d), only connection weights to the tenth hidden neurons remained strong, while all the others became almost zero. When the parameter r increased gradually, the number of stronger connection weights became smaller. Finally, only connection weights into the tenth hidden neurons remained strong. These results show that when the parameter r increased and correspondingly the potential information increased, the number of strong connection weights became smaller.

3.1.3 Self-Assimilated Information and Generalization

The self-assimilated information was computed by keeping the parameter at 0.1. Since the best generalization error was obtained with r=0.1, the parameter r was only changed in the self-assimilation without actual training. Results showed that information could be increased through self-assimilation while keeping generalization errors small. Figure 7 shows the self-assimilated potential information (a) and generalization (b) for four different parameter values. When the parameter values were 0.01 and 0.1, the self-assimilated information increased gradually and in the same direction. When the pa-



(a) Artificial data

(b) Network architecture

Figure 4. Artificial data (a) and network architecture (b)

rameter was 1, the information increased and became larger than 0.6. When the parameter *r* was ten, the self-assimilated information became close to 0.9. On the other hand, the generalization errors were the same as those in Figure 5(b) because the parameter *r* was actually 0.1 in Figure 7(b).

Figure 8 shows connection weights when the parameter *r* increased from 0.01 (a) to 10 (d). The number of stronger connection weights became smaller when the parameter *r* increased. Finally, when the parameter *r* was ten, connection weights into the tenth hidden neuron remained strong. These results show that the self-assimilated information maximization could produce connection weights close to the ones obtained by actually changing the parameter *r* without learning in Figure 6.

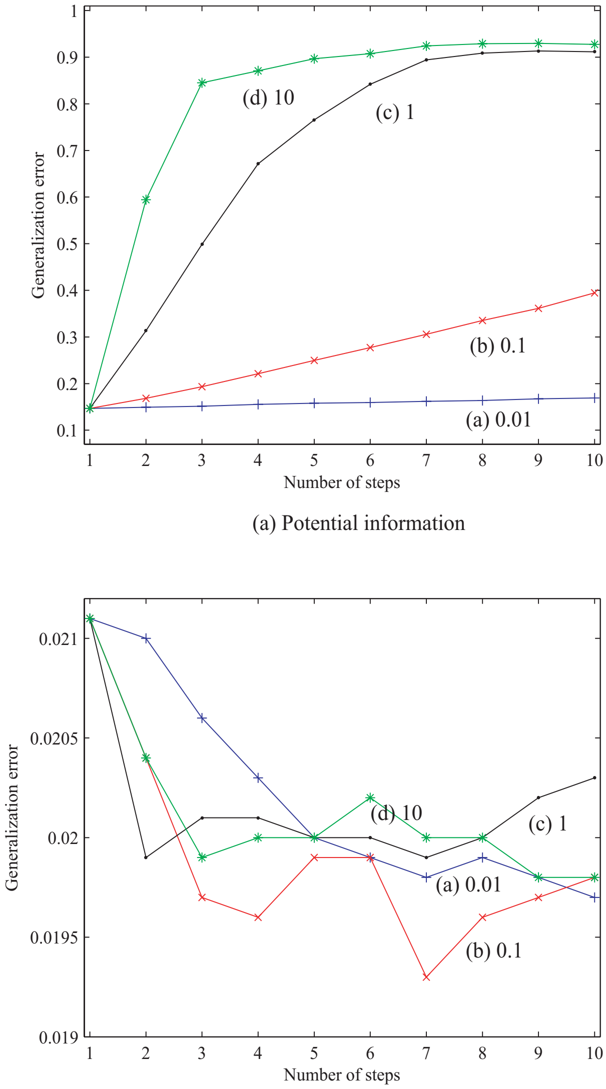
3.1.4 Collective Interpretation

When the parameter *r* increased, the number of strong connection weights became smaller and the same type of connection weights were obtained. Figure 9 shows the self-assimilated connection weights by two parameter values (0.1 and 10) and ten different data sets. As shown in the figures, the strongest connection weights for r=0.1 remained strong for r=10. Even with different data sets and initial conditions, the same type of connection weights could be produced in the end. Thus, it was possible to average the connection weights to examine the characteristics of connection weights.

Figure 10 shows the average self-assimilated connection weights with three different parameter values and by the logistic analysis for the artificial data set. As can be seen in the fugues, the average connection weights from the first input neuron were stronger than those from the second input neuron. When the parameter *r* was 0.1 in Figure 10(a) and 1 in Figure 10(b), the average weights from the first input neuron were much larger than the weights from the second input neuron, but they were negative. When the parameter *r* was 10, in Figure 10(c), the average weights from the second neuron became positive. Thus, the connection weights were close to the regression coefficients by the regression analysis in Figure 10(d). As will be explained later, the generalization error produced by the present method was smaller than that by the regression analysis. In other words, the present method could produce better generalization performance while maintaining the importance of input variables.

3.1.5 Generalization Comparison

The new method could produce the lowest generalization errors with higher potential information as shown in Table 1. When the parameter *r* was 0.01, the potential information was relatively low,



(a) Potential information

(b) Generalization errors

Figure 5. Potential information (a) and generalization errors (b) for the artificial data

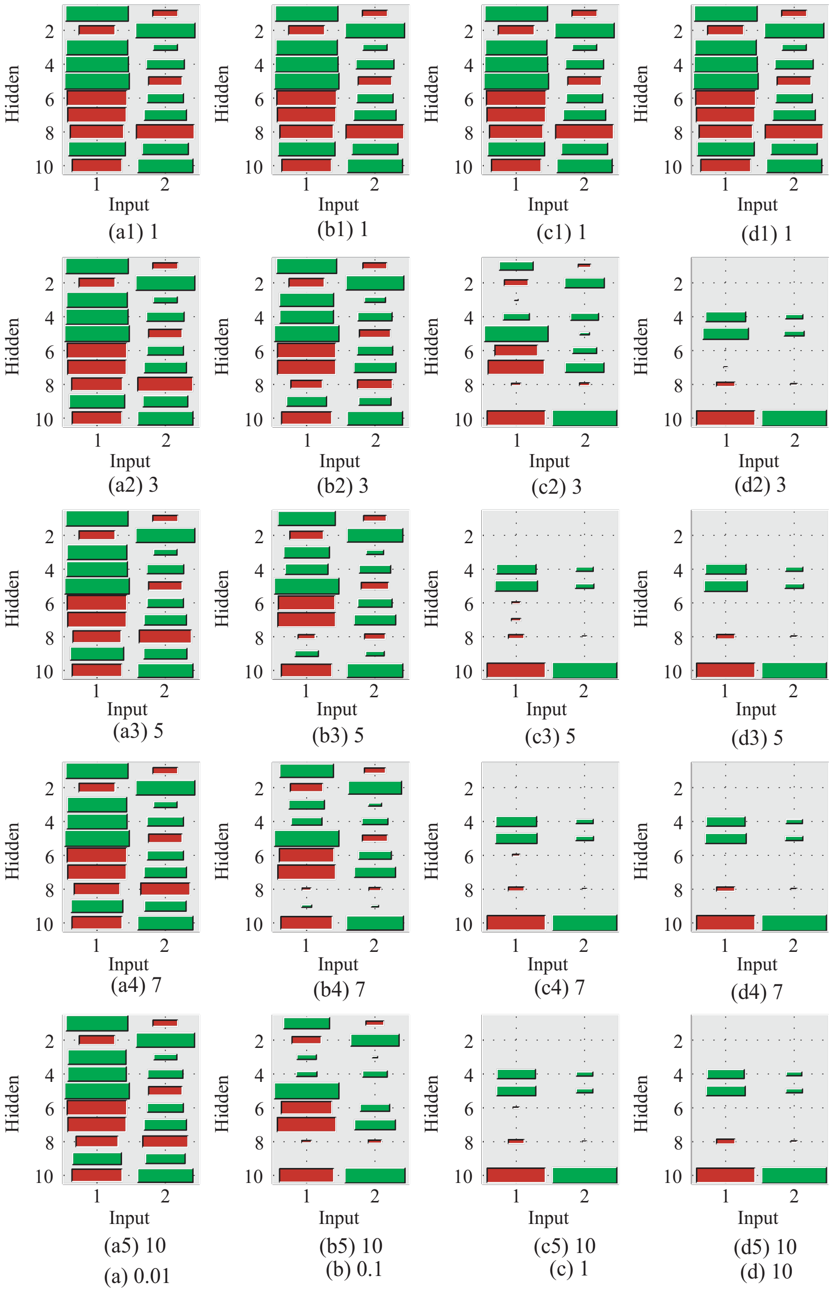
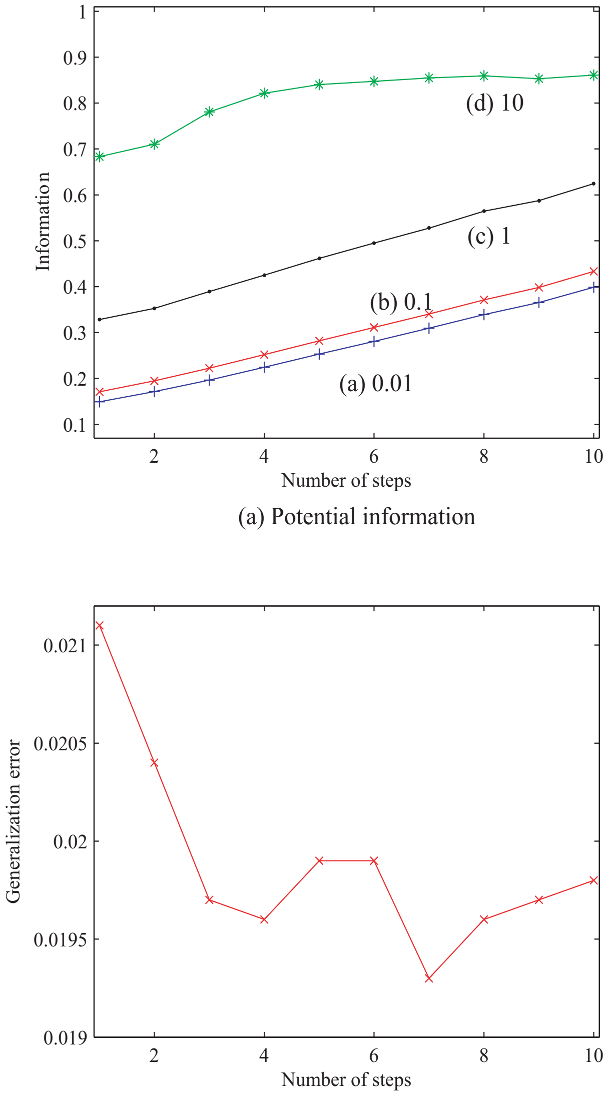


Figure 6. Connection weights by four different parameter values for the artificial data



(a) Potential information

(b) Generalization errors

Figure 7. Self-assimilated potential information (a) and generalization errors (b) by four different parameter values for the artificial data

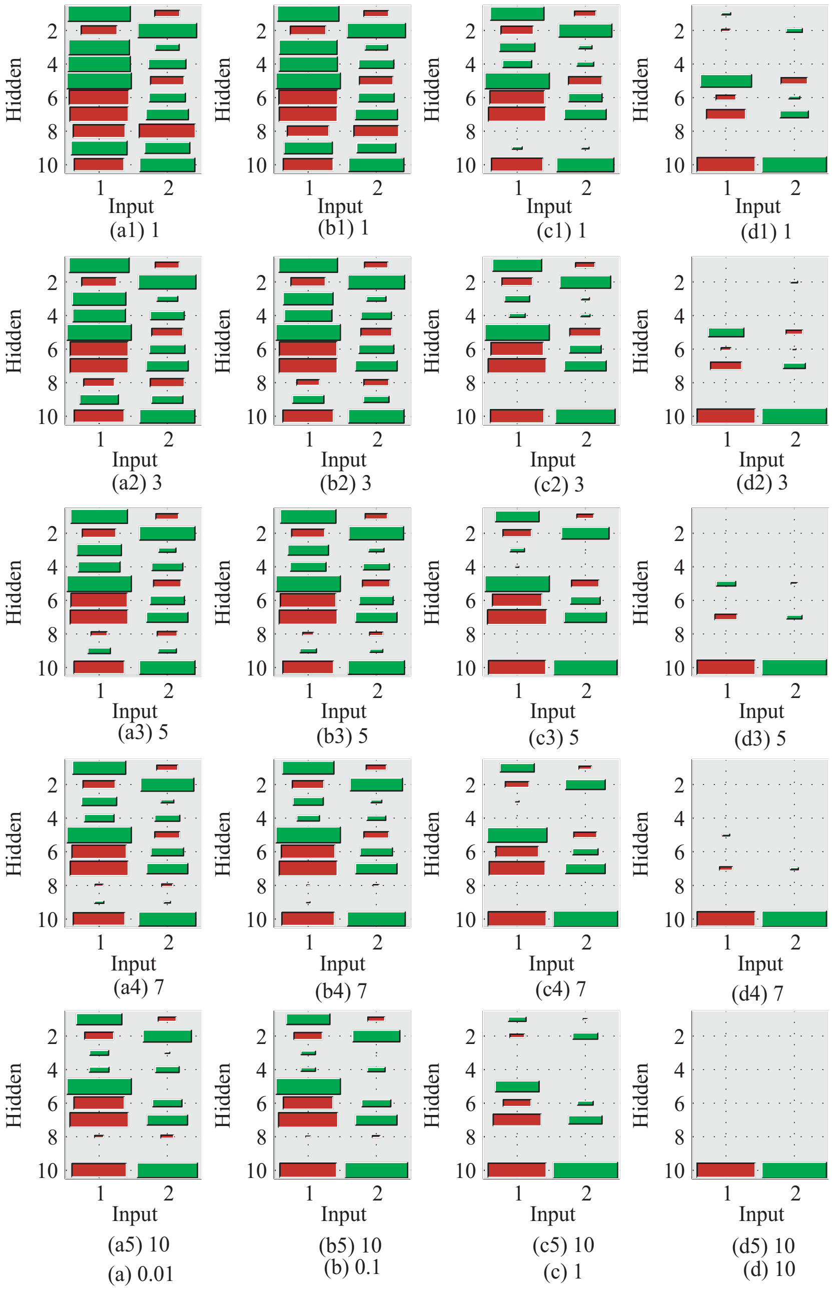
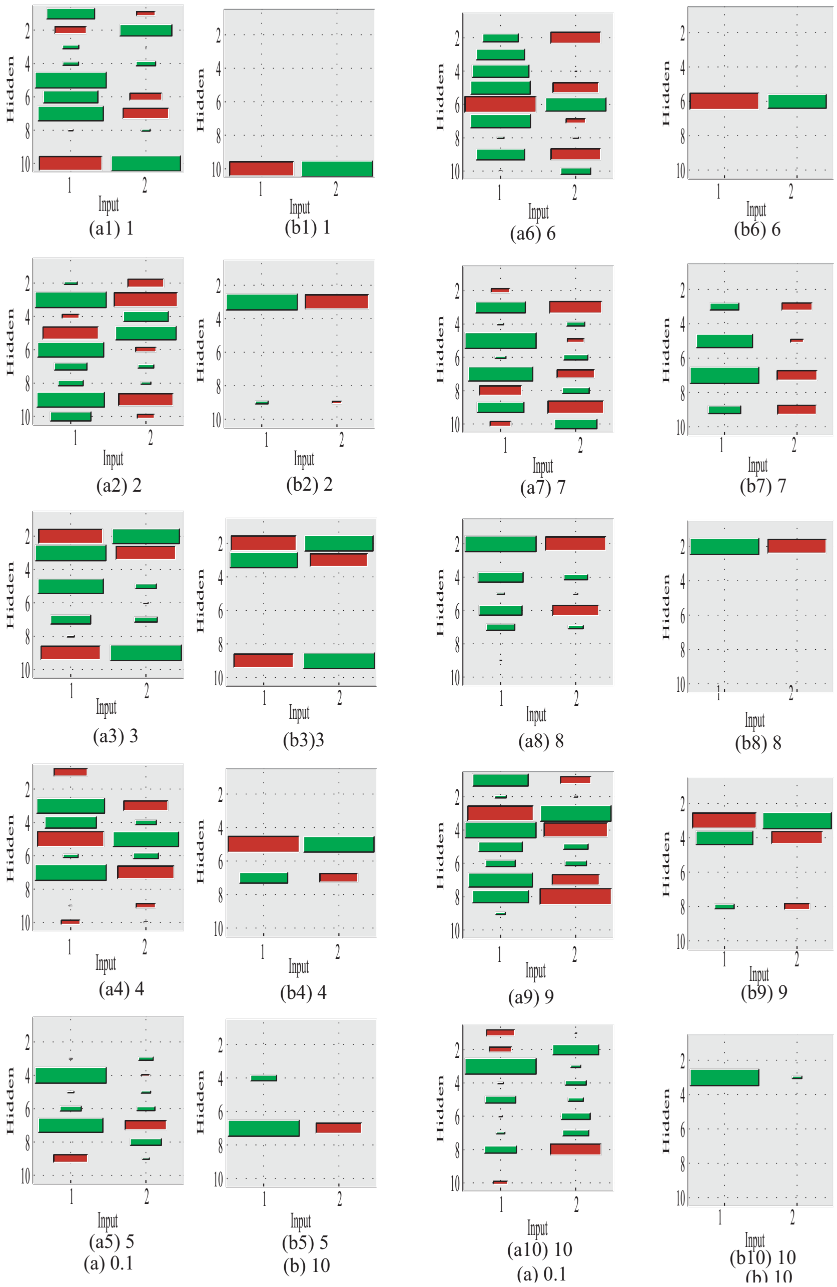


Figure 8. Self-assimilated connection weights by four different parameter values for the artificial data



(a1) 1 (b1) 1 (a6) 6 (b6) 6

(a2) 2 (b2) 2 (a7) 7 (b7) 7

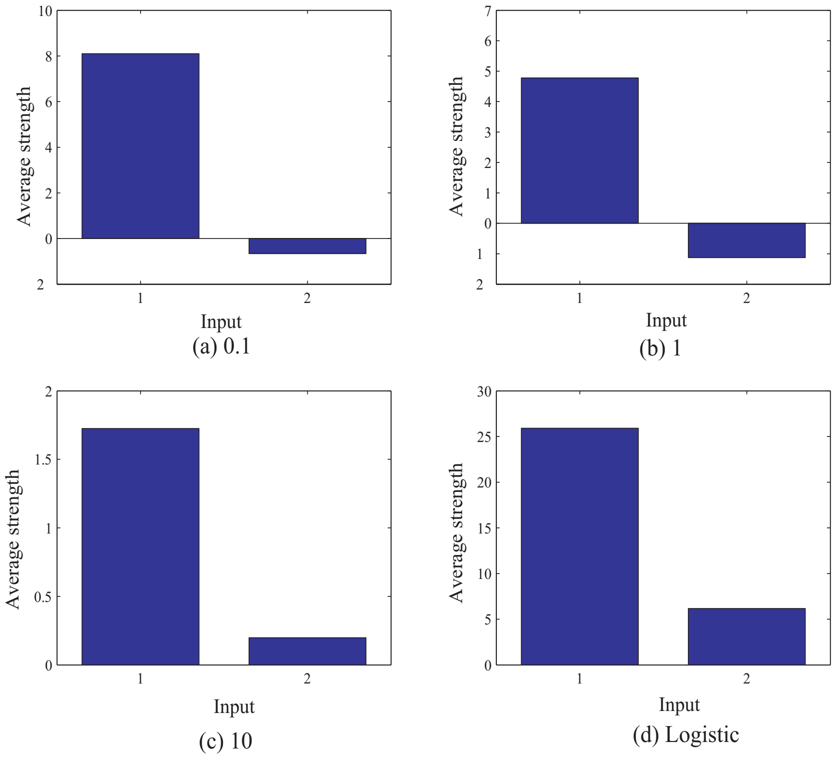
(a3) 3 (b3)3 (a8) 8 (b8) 8

(a4) 4 (b4) 4 (a9) 9 (b9) 9

(a5) 5 (b5) 5 (a10) 10 (b10) 10

(a) 0.1 (b) 10 (a) 0.1 (b) 10

Figure 9. Self-assimilated connection weights by two different parameter values and by ten different initial conditions and different data sets for the artificial data



(a) 0.1

(b) 1

(c) 10

(d) Logistic

Figure 10. Average self-assimilated connection weights by three different parameter values and coefficients by the logistic analysis for the artificial data

namely, 0.1693. When the parameter *r* was increased to 0.1, the potential information increased from 0.1693 to 0.3059. The generalization errors decreased from 0.0197 to the lowest value of 0.0193. Then, when the parameter *r* was further increased to 1 and 10, the potential information was close to 0.9, and the generalization errors increased slightly to 0.0199 and 0.0198. In addition, when the parameter *r* was 0.1, the lowest errors of 0.0193, 0.0180 and 0.0200 were obtained in terms of average, minimum and maximum values. However, compared with the values of 0.0211 (BP) and 0.0202 (Logistic regression), all values were smaller. When self-assimilation was used, the actual parameter in learning was 0.1 and naturally the best generalization error was obtained, as explained above. However, the potential information increased from 0.3096 (r=0.01) to 0.8549(r=10). These results show that self-assimilation could increase potential information while keeping the generalization error low. In other words, self-assimilation could increase information without degrading generalization performance and actual training.

3.2 Counter Services Data

3.2.1 Experimental Outline

This method was also applied to the data on the counter services at a local government office in the Tokyo metropolitan area, provided by the data analysis competition in 2015 organized by the management sciences association[1]. The number of items was 3,194, of which 1,000 were used exclusively for optimizing neural networks. Of these 1,000 items, 70 percent was used for training neural network, while the remaining 30 percent was used for checking the learning. As shown in Figure 11, the number of input, hidden and output neurons were 6, 10 and 2, respectively. The objective of the neural networks was to infer whether the counter services were being interrupted or not. We used Matlab neural networks toolbox with all default values for easy reproduction of the present results.

[1] https://jasmac-j.jimdo.com/

Table 1. Summary of experimental results on generalization performance for the counter services data evaluation data set. The bold face numbers show the best values

Method	*r*	Step	Avg	Std dev	Min	Max	Inf
Potential	0.01	10	0.0197	0.0007	0.0190	0.0210	0.1693
	0.1	7	**0.0193**	0.0007	**0.0180**	**0.0200**	0.3059
	1	7	0.0199	0.0006	0.0190	0.0210	0.8943
	10	9	0.0198	0.0008	0.0190	0.0210	0.9296
Self	0.01						0.3096
	0.1	7	0.0193	0.0007	0.0180	0.0200	0.3403
	1						0.5273
	10						0.8549
BP		1	0.0211	0.0012	0.0200	0.0240	0.1468
Logistic			0.0202	0.0006	0.0190	0.0210	

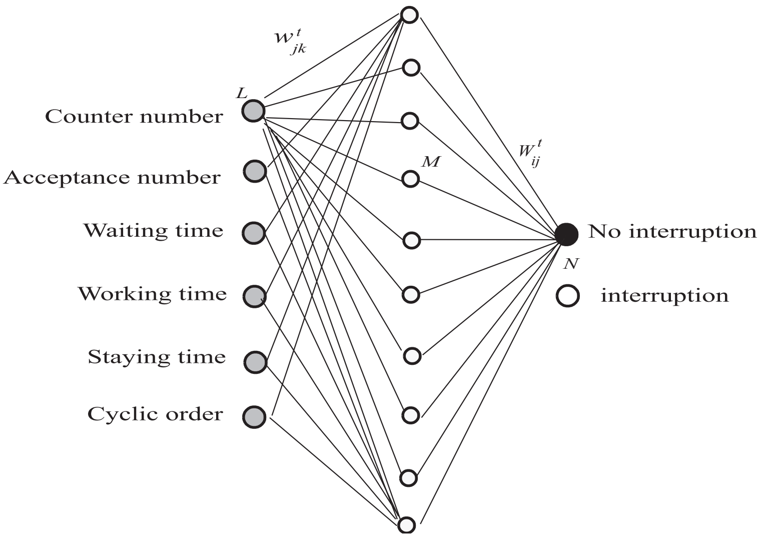


Figure 11. Network architecture for the counter services data

3.2.2 Potential Mutual Information and Generalization

Figure 12(a) shows potential mutual information when the number of steps increased from 1 to 50. When the parameter was 0.01, mutual information increased very slowly and could not go beyond 0.6. When the parameter was 0.1, the information increased immediately to 0.9 with 15 steps and then fluctuated around 0.9. When the parameter was 1, the information jumped to 0.9 with five steps and reached its stable state. When the parameter *r* increased further to 10, the information also jumped to 0.9 with five steps. However, the information began to fluctuate in the later stages of learning steps. The results show that by increasing the parameter *r*, the information could be increased.

Figure 12(b) shows generalization errors as a function of the number of steps. When the parameter was 0.01, the errors decreased gradually and reached their lowest value with around 30 steps. When the parameter *r* was 0.1, the generalization errors decreased gradually until the number of steps was 25. Then, the error could not be decreased. When the parameter *r* increased further to 1 and 10, the errors sharply decreased only with a few learning steps. Then, the errors began to fluctuate around 0.09. The results show that the generalization errors could not be decreased, when the parameter *r* was increased. When the parameter *r* increased, the potential information tended to increase as shown in Figure 12(a). This means that when the parameter increased, too much information or excessive information was accumulated. This excessive information tends to degrade generalization performance, making it necessary to reduce this excessive information content.

3.2.3 Connection Weights

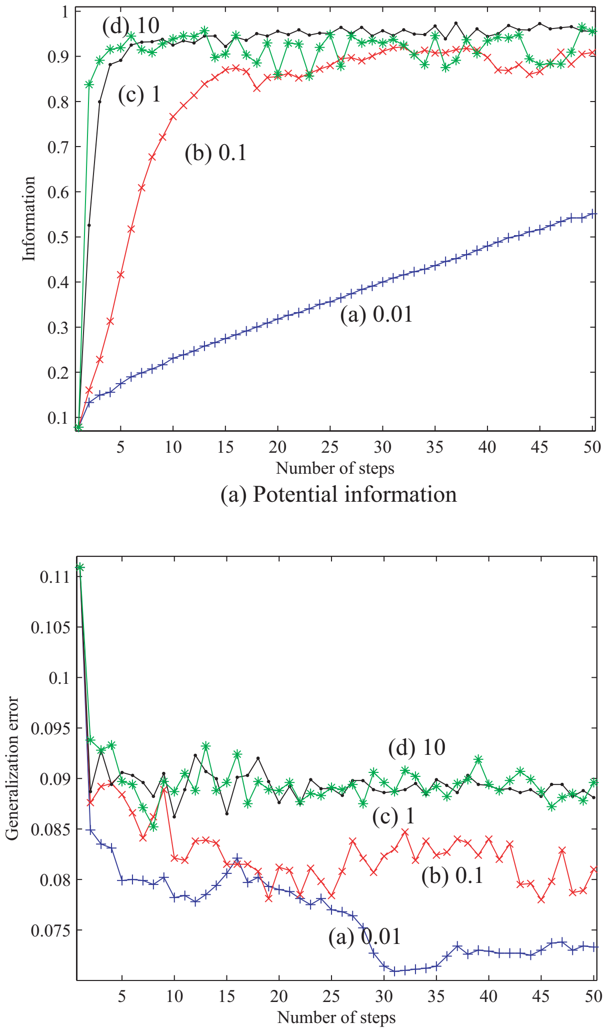
Figure 13 shows connection weights when the parameter increased from 0.01 to 10. As can be seen in the figure, when the parameter increased, gradually, connection weights into the ninth hidden neurons remained strong, while all the other weights were pushed toward zero. When the parameter *r* was 0.01, many strong connection weights were produced. Though connection weights into the ninth hidden neuron became stronger, we had some difficulty in detecting important connection weights. When the parameter was increased to 0.1, the connection weights into the ninth hidden neuron became more explicit. When the parameter was increased from 1 and 10, for all steps except the first step, the connection weights into the ninth hidden neuron remained strong, while all the other connection weights became almost zero.

The results show that if we try to improve generalization performance, we need heavily distributed connection weights, which prevents us from interpreting the connection weights. On the other hand, if we try to interpret connection weights, we need to increase the parameter *r* and hence increase the potential information. This increase in the potential information has the effect of simplifying connection weights for better interpretation. However, generalization performance is degraded because of the excessive potential information.

3.2.4 Self-Assimilated Information and Generalization

To reduce the excessive information content, we introduced self-assimilation, where the characteristics of connection weights are used to modify the weights themselves. We expected that the characteristics of weights would be accentuated by the process of self-assimilation. Thus, with a relatively small number of learning steps, we presumed that we could estimate the main characteristics of the connection weights. For the parameter *r*, keeping the parameter *r* smaller, we attempted to extract connection weights similar to those obtained when the parameter *r* was much larger.

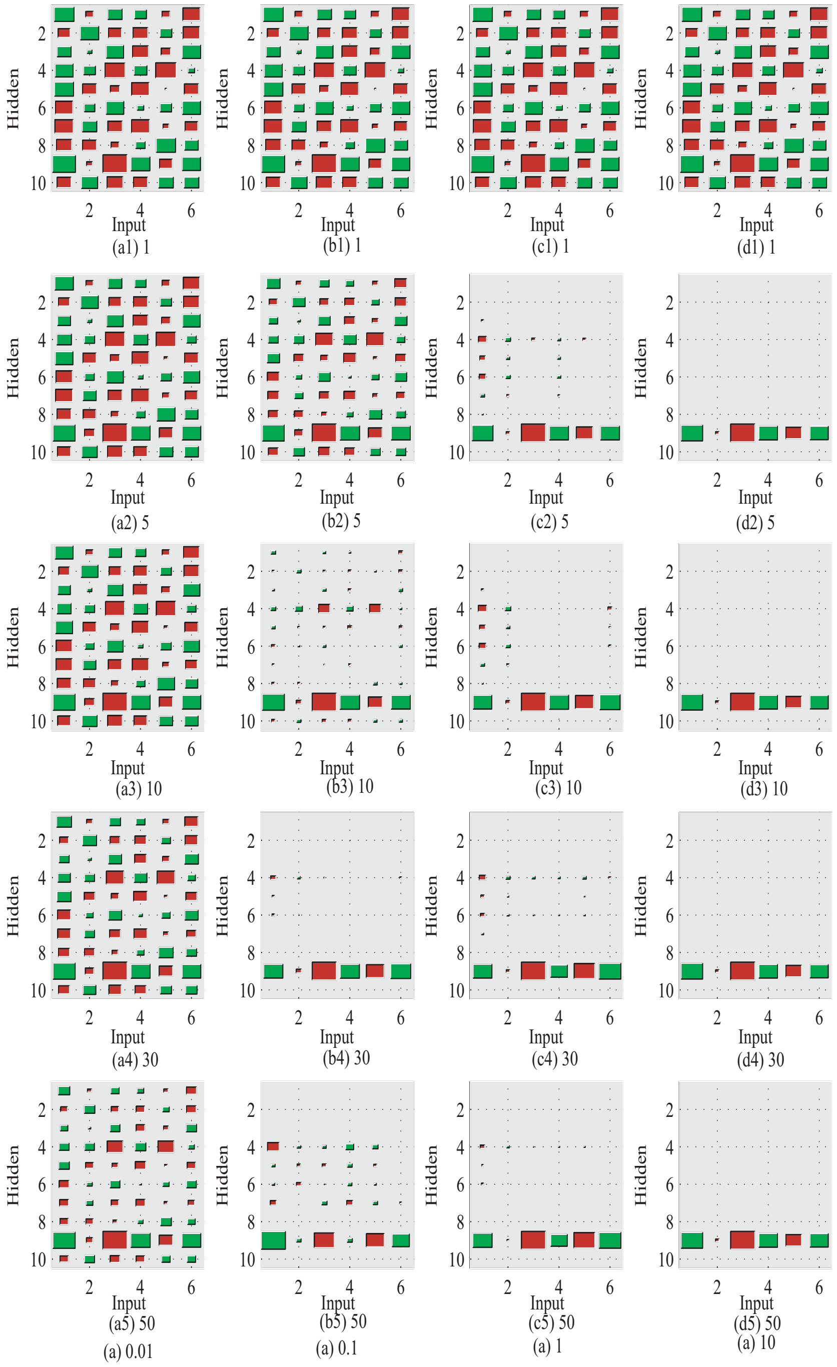
Figure 14 shows self-assimilated information when the parameter increased from 0.01 to 10. As shown in Figure 14(b), generalization errors were lower because the parameter was actually set to 0.01. However, potential mutual information was increased in this case. As shown in Figure 14(a), when the parameter *r* was 0.01, the potential information slowly increased but to a level below 0.6. When the parameter was 0.1, the information increased over 0.6. When the information was further increased to 1, the information became close to 0.9. Finally, when the parameter *r* was 10, the information increased to almost the maximum value



(a) Potential information

(b) Generalization errors

Figure 12. Potential mutual information (a) and generalization errors (b) for four different parameter values: 0.01 (a), 0.1 (b), 1 (c) and 10 (d) for the counter services data.



(a1) 1 (b1) 1 (c1) 1 (d1) 1

(a2) 5 (b2) 5 (c2) 5 (d2) 5

(a3) 10 (b3) 10 (c3) 10 (d3) 10

(a4) 30 (b4) 30 (c4) 30 (d4) 30

(a5) 50 (b5) 50 (c5) 50 (d5) 50

(a) 0.01 (a) 0.1 (a) 1 (a) 10

Figure 13. Connection weights by four different parameter values for the counter services data.

of 1, in just three steps. This shows that the self-assimilation can be used to increase the information or the self-assimilated information almost to the maximum value of 1 in the end, while maintaining good generalization performance.

3.2.5 Self-Assimilated Connection Weights

Figure 15 shows the connection weights when the parameter increased from 0.01 to 10. When the parameter increased, the connection weights into the ninth hidden neuron became stronger. When the parameter r was 0.01, many strong connection weights were produced. When the parameter r increased from 0.01 to 0.1, connection weights into the ninth neuron tended to gradually become larger. When the parameter r was 1, connection weights into the ninth hidden neuron became clearer, while keeping some weak connection weights. When the parameter was ten, connection weights into the ninth hidden neuron remained strong while all the other connection weights became almost zero.

If we compare the connection weights in Figure 13 with the self-assimilation weights in Figure 15, we can see that the self-assimilated weights clearly represented the real weights. Because the self-assimilated weights are not accompanied by learning processes, it can be that self-assimilation is able to estimate the characteristics of weights in a much simpler manner.

3.2.6 Collective Interpretation

Figures 16(a) and (b) show connection weights adjusted by the signs of hidden-output connection weights by ten different data sets and initial conditions for r=0.01 and 10, respectively. When the parameter r was 0.01, connection weights to a specific neuron became larger, but there were still many weak connection weights. When the parameter r increased to 10, those weak connection weights disappeared and connection weights into a specific neuron remained strong. Of ten connection weights, six weights showed that the connection weights from the first neuron were much clearer than the other ones. Thus, we can see that the first input neuron played the most important role in learning.

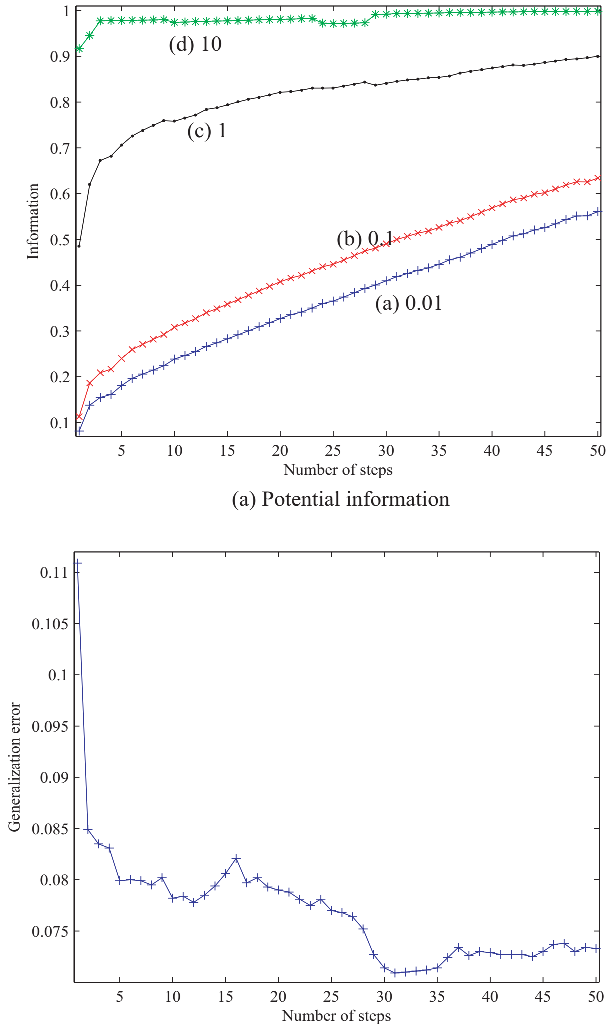
Figure 17(a) shows the average connection weights when the parameter was 0.01. Connection weights from the first, fourth and sixth input neurons were strongly positive, while weights from the third and fifth input neurons were strongly negative. When parameter increased to 10 in Figure 17(b), connection weights from the first input neuron remained strong, while all the other connection weights became smaller. Figure 17(c) shows regression coefficients by the logistic regression analysis. One of the major differences between the methods lay in the strength of the connection weights from the first input neuron, which was much smaller than those by the potential information method in Figure 17(a) and (b). In other words, self-assimilation was able to highlight the importance of the first input neuron clearly.

Let us interpret the average weights in Figure 17(b). The connection weights from the third input neuron and the fifth input neuron, representing the waiting time and staying time, were strongly negative. This means that as the waiting time and staying time become larger, the possibility of interruption became naturally higher. On the other hand, the larger positive connection weight from the first input neuron could be observed. The first input neuron was simply the counter number. In other words, when the counter number increased, the possibility of interruption became smaller. This suggests that the counter placement in the city government office was a major cause of interruption. Practically, what this means is that to reduce the loss by the interruption, it may be wise to re-organize the placement of the counters.

3.2.7 Generalization Comparison

Table 2 shows generalization comparison by the present method, BP with the early stopping and the logistic analysis. The lowest average error of 0.071 was obtained by the present method. The lowest errors of 0.043 and 0.093 in terms of minimum and maximum error were also obtained by the present method. Thus, the lowest errors in terms of the average, minimum and maximum values were obtained by the present method. By the logistic analysis, the error increased to 0.101. Finally, the worst error of 0.111 was obtained by the BP with the early stopping.

For the standard deviation, the smallest value of 0003 was obtained by the logistic regression analysis. This means that the stable solutions were obtained by the logistic regression analysis. By the



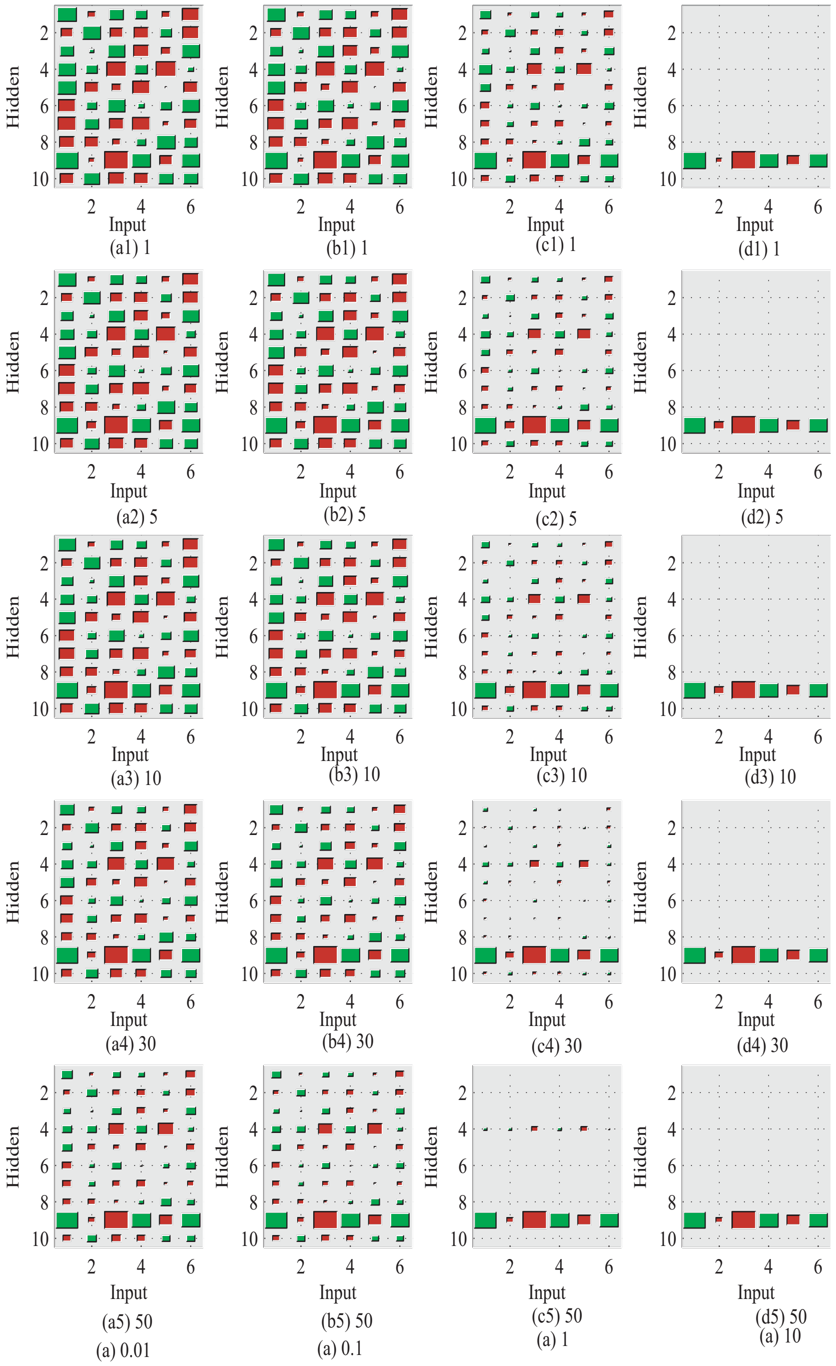
(a) Potential information

(b) Generalization errors

Figure 14. Self-assimilated information (a) and generalization errors (b) for different parameter values for the counter services data.

Table 2. Summary of experimental results on generalization performance for the counter services data evaluation data set. The bold face numbers show the best values.

Method	*r*	Step	Avg	Std dev	Min	Max	Inf
Potential	0.01	31	**0.071**	0.017	**0.043**	**0.093**	0.409
	0.10	45	0.078	0.016	0.056	0.101	0.866
	1	10	0.086	0.016	0.052	0.101	0.925
	10	8	0.085	0.013	0.061	0.101	0.908
Self	0.01	31	**0.071**	0.017	**0.043**	**0.093**	0.419
	0.10	31					0.500
	1	31					0.845
	10	31					0.993
BP		1	0.111	0.048	0.076	0.242	0.078
Logistic			0.101	**0.003**	0.096	0.107	



(a1) 1 (b1) 1 (c1) 1 (d1) 1

(a2) 5 (b2) 5 (c2) 5 (d2) 5

(a3) 10 (b3) 10 (c3) 10 (d3) 10

(a4) 30 (b4) 30 (c4) 30 (d4) 30

(a5) 50 (b5) 50 (c5) 50 (d5) 50

(a) 0.01 (a) 0.1 (a) 1 (a) 10

Figure 15. Self-assimilated weights by four different initial weights for the counter services data.

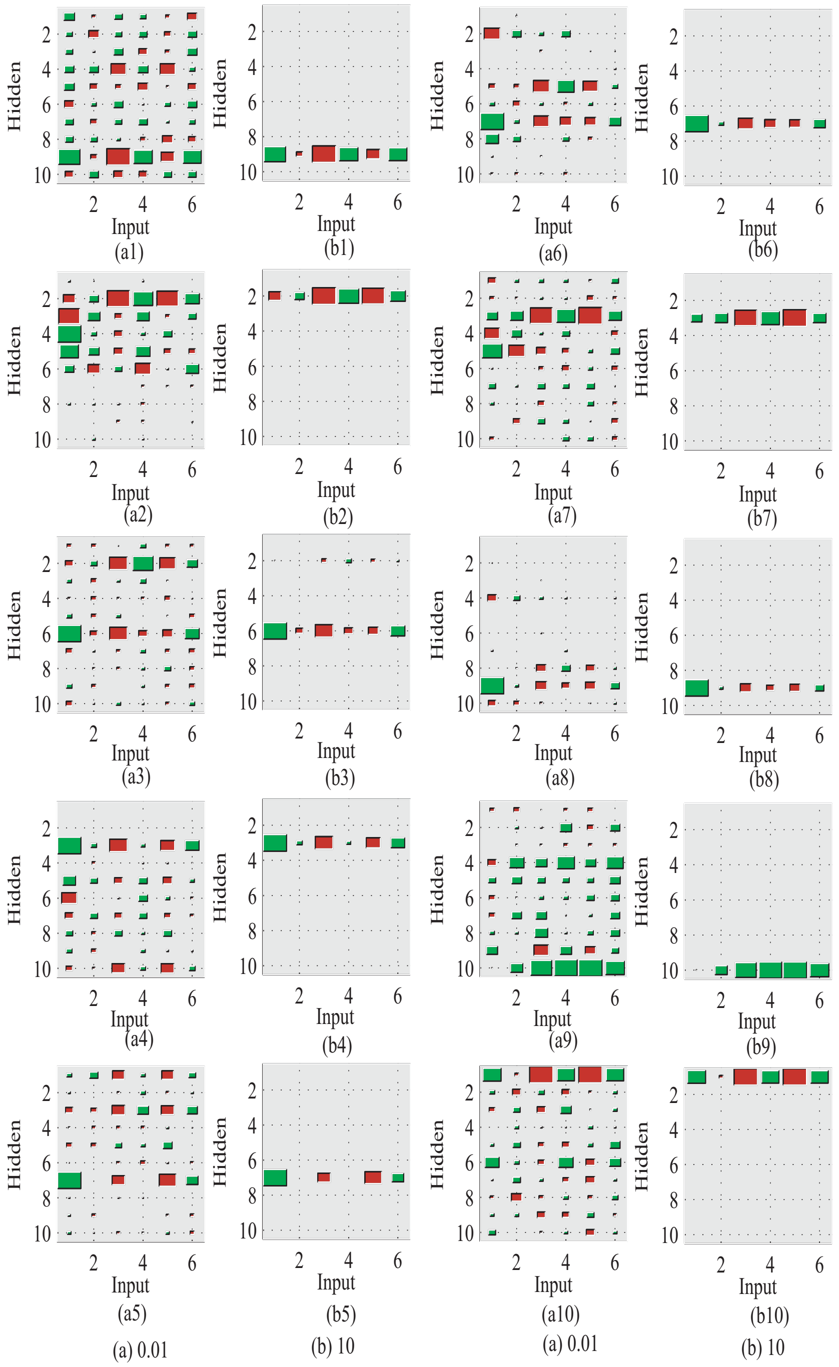


Figure 16. Self-assimilated weights multiplied by sign$W_{1j}$with $r = 0.01$ (a) and $r = 10$ (b) by ten different data sets and initial conditions for the counter services data.

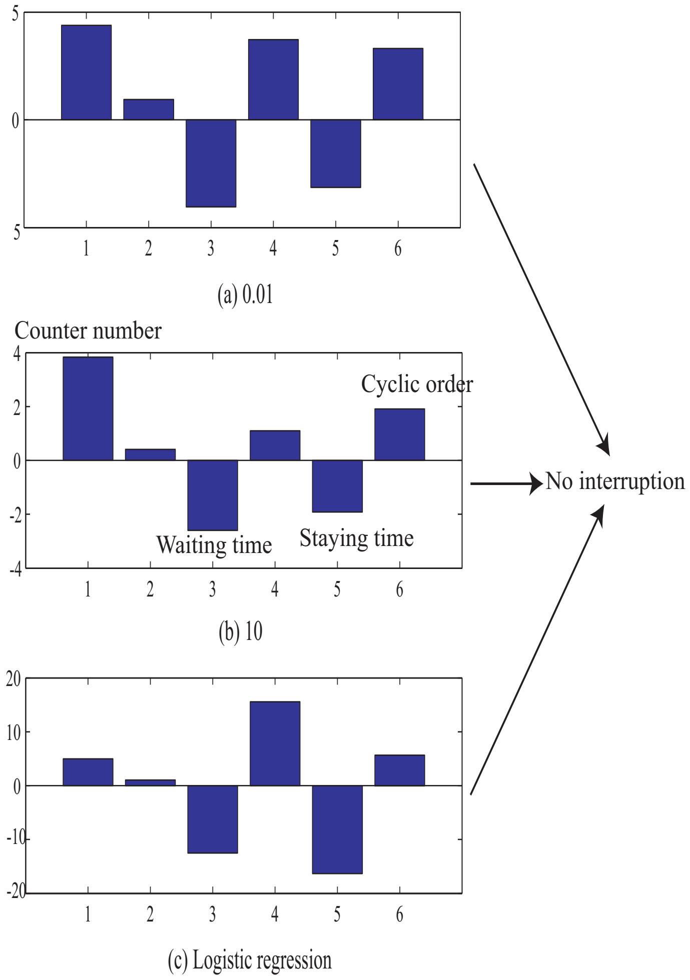


Figure 17. Average weights and regression coefficients by potential learning with $r = 0.01$ (a) and $r = 10$ (b) and the regression analysis (c) for the counter services data.

conventional BP with the early stopping, the standard deviation was 0.048 - the largest value. When the parameter r increased from 0.01 to 10, the standard deviation decreased from 0.017 to 0.013. This means that by increasing information, the learning became more stable.

Finally, the real information was 0.078 by the conventional BP. When the parameter increased from 0.01 to 1, the information increased from 0.409 to 0.925. Then, the information decreased slightly to 0.908, when the parameter r was 10. Correspondingly, the generalization error increased from 0.071 to 0.086 when the parameter increased from 0.01 to 1. Then, the average error decreased slightly to 0.085 when the parameter r was 10. This suggests that the excessive information acquisition degrades generalization performance.

4 Conclusion

The present paper proposed a new type of information-theoretic method called "self-assimilation" to solve the problem of excessive information acquisition. Thus far, the potential learning method has shown good performance in terms of generalization and interpretation. However, one of the major problems is that the method tends to acquire information content excessively in the early stages of learning. To cope with this problem, we had previously attempted to make learning as slow as possible by reducing the potentiality parameter. This means that neural networks tend to acquire information content very slowly, a require a large number of learning steps.

To solve this problem of excessive information acquisition, we introduced the self-assimilation method where the characteristics of connection weights are enhanced by increasing the potentiality parameter. This makes it possible to predict the future characteristics of connection weights. Thus, we can improve generalization performance by eliminating excessive information and at the same time interpreting connection weights whose future characteristics are predicted by the enhancement of the weights.

The method was applied to the data of the counter services of a local government of Tokyo metropolitan area. The results showed that excessive information was eliminated, and in addition, a smaller number of connection weights were produced for better interpretation. One of the major problems is how to choose the appropriate potentiality parameter to compromise between the ratio of learning and generalization. However, even at the present stage of study, it can be said that the method is simple enough to be applied to large-scale data and multi-layered neural networks.

Acknowledgement

We would like to offer my special thanks to the management sciences association[2]. The data on the counter services of a local government of Tokyo metropolitan area was actually provided by the data analysis competition in 2015 organized by the association. This research is supported by the Japan Society for the Promotion of Science under the Grants-in-Aid for Scientific Research-grant 16K00339.

References

[1] R. Linsker, Self-organization in a perceptual network, Computer, vol. 21, no. 3, pp. 105–117, 1988.

[2] R. Linsker, How to generate ordered maps by maximizing the mutual information between input and output signals, Neural computation, vol. 1, no. 3, pp. 402–411, 1989.

[3] R. Linsker, Local synaptic learning rules suffice to maximize mutual information in a linear network, Neural Computation, vol. 4, no. 5, pp. 691–702, 1992.

[4] R. Linsker, Improved local learning rule for information maximization and related applications, Neural networks, vol. 18, no. 3, pp. 261–265, 2005.

[5] G. Deco, W. Finnoff, and H. Zimmermann, Unsupervised mutual information criterion for elimination of overtraining in supervised multilayer networks, Neural Computation, vol. 7, no. 1, pp. 86–107, 1995.

[6] G. Deco and D. Obradovic, An information-theoretic approach to neural computing, Springer Science & Business Media, 2012.

[7] H. B. Barlow, Unsupervised learning, Neural computation, vol. 1, no. 3, pp. 295–311, 1989.

[2] https://jasmac-j.jimdo.com/

[8] H. B. Barlow, T. P. Kaushal, and G. J. Mitchison, Finding minimum entropy codes, Neural Computation, vol. 1, no. 3, pp. 412–423, 1989.

[9] J. J. Atick, Could information theory provide an ecological theory of sensory processing?, Network: Computation in neural systems, vol. 3, no. 2, pp. 213–251, 1992.

[10] Z. Nenadic, Information discriminant analysis: Feature extraction with an information-theoretic objective, Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 29, no. 8, pp. 1394–1407, 2007.

[11] J. C. Principe, D. Xu, and J. Fisher, Information theoretic learning, Unsupervised adaptive filtering, vol. 1, pp. 265–319, 2000.

[12] J. C. Principe, Information theoretic learning: Renyi's entropy and kernel perspectives, Springer Science & Business Media, 2010.

[13] K. Torkkola, Feature extraction by non parametric mutual information maximization, The Journal of Machine Learning Research, vol. 3, pp. 1415–1438, 2003.

[14] R. Kamimura, Simple and stable internal representation by potential mutual information maximization, in International Conference on Engineering Applications of Neural Networks, pp. 309–316, Springer, 2016.

[15] R. Kamimura, Self-organizing selective potentiality learning to detect important input neurons, in Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on, pp. 1619–1626, IEEE, 2015.

[16] R. Kamimura, Collective interpretation and potential joint information maximization, in Intelligent Information Processing VIII: 9th IFIP TC 12 International Conference, IIP 2016, Melbourne, VIC, Australia, November 18-21, 2016, Proceedings, pp. 12–21, Springer, 2016.

[17] R. Kamimura, Repeated potentiality assimilation: Simplifying learning procedures by positive, independent and indirect operation for improving generalization and interpretation (in press), in Proc. of IJCNN-2016, (Vancouver), 2016.

[18] R. Kamimura and T. Kamimura, Structural information and linguistic rule extraction, in Proceedings of ICONIP, pp. 720–726, 2000.

[19] R. Kamimura, T. Kamimura, and O. Uchida, Flexible feature discovery and structural information control, Connection science, vol. 13, no. 4, pp. 323–347, 2001.

[20] R. Kamimura, Information-theoretic competitive learning with inverse euclidean distance output units," Neural processing letters, vol. 18, no. 3, pp. 163–204, 2003.

Ryotaro Kamimura is currently a professor of IT Education Center of Tokai University in Japan. His research interests are information-theoretic approach to neural computing.

Tsubasa Kitago was a student of School of Political Science and Economics of Tokai University in Japan. His research interests are the artificial intelligence and its application to data analysis and autonomous cars.