Neurocomputing 207 (2016) 676–683

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

# Mutual information based weight initialization method for sigmoidal feedforward neural networks

Junfei Qiao [a,b,*], Sanyi Li [a,b], Wenjing Li [a,b]

[a] College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100124, China
[b] Beijing Key Laboratory of Computational Intelligence and Intelligent System, Beijing 100124, China

ARTICLE INFO

*Article history:*
Received 27 January 2016
Received in revised form
7 April 2016
Accepted 24 May 2016
Communicated by Ma Lifeng Ma
Available online 4 June 2016

*Keywords:*
Sigmoidal feedforward neural network
Weight initialization
Mutual information

ABSTRACT

When a sigmoidal feedforward neural network (SFNN) is trained by the gradient-based algorithms, the quality of the overall learning process strongly depends on the initial weights. To improve the algorithm stability and avoid local minima, a Mutual Information based weight initialization (MIWI) method is proposed for SFNN. The useful information contained in input variables is measured with the mutual information (MI) between input variables and output variables. The initial distribution of weights is consistent with the information distribution in the input variables. The lower and upper bounds of the weights range are calculated to ensure the neurons inputs are within the active region of sigmoid function. The MIWI method makes the initial weights close to the global optimal point with a higher probability and avoids premature saturation. The efficiency of the MIWI method is evaluated based on several benchmark problems. The experimental results show that the stability and accuracy of the proposed method are better than some other weight initialization methods.



## 1. Introduction

Neural network can be considered as a nonlinear system and the training process can be seen as a nonlinear optimization process [1,2]. Due to the fact that the optimal weights are difficult to be found with analytical methods, iterative local or global optimization methods are necessary [3]. The gradient-based (GB) training algorithms are widely used due to its effectiveness [3]. However, GB methods have slow convergence rate and are often hampered by the occurrence of local minima [4]. It has been demonstrated that the performance (convergence rate and training accuracy) of sigmoidal feedforward neural network (SFNN) trained with GB algorithms depends on several factors, including training algorithm, initial conditions, training data and network structure [5–8]. Use of appropriate weight initialization can shorten the training time and avoid the local minima caused by random initial weights [9–22].

Many methods have been developed to choose the suitable initial weights of SFNN. Generally speaking, they can be classified into two categories, namely, least squares (LS) methods and interval analysis (IA) methods. The LS methods calculate accurate initial weights to diminish the initial error, which has been employed in many literatures. Yam [9] proposed a weight initialization algorithm based on a linear algebraic, and the optimal initial weights of each layer are evaluated by LS method. Erdogmus [10] proposed a backpropagation of the desired response algorithm that approximates the nonlinear least squares problem with linear least squares. Erdogmus [3] improved the optimization accuracy by further considering the local scaling effects of the nonlinearities. Liu [11] used the partial LS algorithm to set the initial weights and the optimal number of hidden neurons simultaneously. Timotheou [1] approximated the nonlinear equations of the network to obtain linear equations with nonnegativity constraints, and then developed a projected gradient algorithm to solve the formulated linear nonnegative LS problem. These algorithms can diminish the initial error effectively, but cannot avoid local minima and the performances of those methods are unstable.

The IA methods determine optimal range for the initial weights and biases to ensure the hidden neurons be active or avoid premature saturation. The optimal range is investigated from different aspects. Drago [20] obtained the maximum magnitude of the weights based on statistical analysis to improve the convergence speed. Thimm [15] designed many experiments to determine the optimal range for the initial weights and biases of high-order networks. Yam [18] determined the initial weights based on the Cauchy's inequality and proposed a linear algebraic method to guarantee the outputs of the neurons within the active region. In subsequent research, Yam [19] proposed a method based on multidimensional geometry, which ensured the activation function be fully utilized.

* Corresponding author at: College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100124, China.
*E-mail address:* junfeiq@bjut.edu.cn (J. Qiao).

http://dx.doi.org/10.1016/j.neucom.2016.05.054
0925-2312/

Yang [22] proposed a weight initialization method based on the theory that the range of the initial values should be greater than the adjustment quantities. Adam [21] proposed a linear interval tolerance approach to guarantee the input of each hidden neuron be in the active region of sigmoid function. Talaska [17] proposed an initialization mechanism based on a Convex Combination method that is easy to be realized in transistor. Although the convergence rate with these methods can be accelerated, the performance is unstable and cannot avoid local minima.

In order to solve the stability problem, we propose a Mutual Information based weight initialization (MIWI) method in this paper, which takes advantage of mutual information (MI) in evaluation of information between variables. MI has been demonstrated to be effective in neuron network to measure the interrelation of two neurons. For example, Qiao [23] added or removed hidden neurons of RBF neural network based on the MI between hidden neurons and output neurons. Zhang [24] proposed an adaptive merging and splitting algorithm for feedforward neural networks based on MI. Peng [25] proposed the minimal-redundancy-maximal-relevance criterion based on MI to select optimal hidden neurons. Puma-Villanueva [44] proposed a constructive algorithm for feedforward neural network based on MI. Chen [26] proposed a clustering algorithm based on partial mutual information (PMI) to implement clustering neurons. Besides some other researchers [27–29] also applied PMI to selecting input variables of artificial neural network.

In this paper, the MIWI algorithm adopts the MI between input variables and output variables to measure the useful information contained in input variables (the neurons with high MI contain more useful information). The MIWI is mainly based on three theorems that have been verified:

(1) If the initial weights are close to the global optimum, any descent algorithm can train the weights toward the optimum reliably [3];
(2) The network with optimal weights can fully reflect the relationship between the input variables and output variables [17];
(3) Restrain the range of initial weights reasonably can shorten the training time dramatically [16,21].

The MIWI algorithm has several advantages as follows: Firstly, MI is calculated between each input variable and output variable. Input variable with high MI means the information contained is more important. Since the sigmoid function is a monotone increasing function, the initial weights between input neurons and hidden neurons are positively related to the mutual information of the input neuron they connected. Thus the initial distribution of the weights is consistent with the information distribution in the input variables. To ensure the diversity of hidden neurons, the biases of hidden neurons are distributed in the value space uniformly and randomly. Thus the initial weights have higher probability to get close to the optimal point and the performance can be more stable than by random initialization.

Secondly, based on the distribution of the initial weights, the lower and the upper bounds of the initial weights are calculated to guarantee all the hidden neurons are active in the initial phase. Then the convergence rate can be guaranteed and the premature saturation can be avoided.

This paper is arranged as follows. The basic conceptions are introduced in Section 2. The MIWI algorithm is described particularly in Section 3. The experimental results and the performance comparison between MIWI and other weight initialization methods are presented in Section 4. A discussion of the merits of the proposed MIWI algorithm is given in Section 5. The conclusion is presented in Section 6. For convenience of discussion, the acronyms used in this paper are listed in Table 1.

**Table 1**
Lists of acronyms.

| Acronym | Description |
|---|---|
| SFNN | Sigmoidal feedforward neural network |
| BP | Backpropagation |
| GB | Gradient-based |
| MIWI | Mutual Information based weight initialization |
| MI | Mutual information |
| LS | Least squares |
| IA | Interval analysis |
| PMI | Partial mutual information |
| MLPs | Multilayer perceptions |
| SCBP | Split-complex backpropagation algorithm |
| ETP | Effluent total phosphorus |
| T | Temperature |
| ORP | Oxidation reduction potential |
| DO | Dissolved oxygen concentration |
| TSS | Total soluble solid |
| pH | Potential of hydrogen |
| ITP | Influent total phosphorus |

## 2. Basic conceptions

In this section, we briefly introduce SFNN and MI.

### 2.1. Sigmoidal feedforward neural networks (SFNN)

The SFNN is a kind of multilayer perceptions (MLPs) that applies sigmoidal activation functions in hidden neurons. Due to the MLPs with one hidden layer can approximate any continuous function [30,31], we mainly research the one-hidden-layer SFNN. The structure of a one-hidden-layer SFNN is shown in Fig. 1.

In this paper, let $n_0$,$n_1$, and$n_2$denote the number of neurons in the input layer, the hidden layer and the output layer respectively. The input and output of the three layers are $[z^{(0)},y^{(0)}]$, $[z^{(1)},y^{(1)}]$ and $[z^{(2)},y^{(2)}]$, respectively. Assume the input vector is $X=[x_1,x_2,\ldots,x_{n_0}]$, then the output of the $i$th input neuron can be expressed as

$$y_i^{(0)}=x_i,(i=1,2,\ldots,n_0) \tag{1}$$

In hidden layer, each neuron is connected to all the input neurons. The input and output of the $j$th hidden neuron are

$$z_j^{(1)}=\sum_{i=1}^{n_0} w_{ij}^{(1)}y_i^{(0)}+b_j^{(1)},(j=1,2,\ldots,n_1) \tag{2}$$

$$y_j^{(1)}=f^{(1)}\left(z_j^{(1)}\right)=\left(1+\exp\left(-z_j^{(1)}\right)\right)^{-1} \tag{3}$$

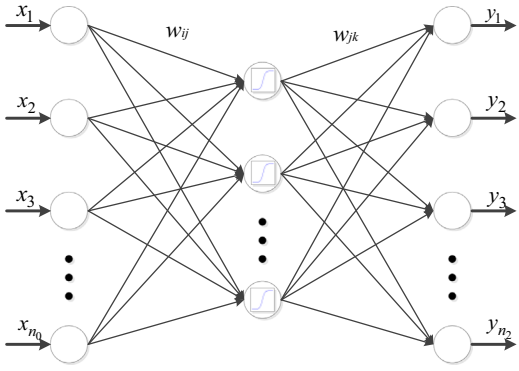


**Fig. 1.** The one-hidden-layer SFNN structure.

where $z_j^{(1)}$is the input value of the $j$th hidden neuron; $w_{ij}^{(1)}$ is the weight between the $i$th input neuron and the $j$th hidden neuron; $b_j^{(1)}$ is the bias of the $j$th hidden neuron; $y_j^{(1)}$ is the output of the $j$th hidden neuron. The activation function of hidden neurons $f^{(1)}(.)$ is Log-Sigmoidal function.

The input and output of the $k$th output neuron can be expressed as

$$z_k^{(2)} = \sum_{j=1}^{n_1} w_{jk}^{(2)} y_j^{(1)} + b_k^{(2)}, (k = 1, 2, \ldots, n_2) \tag{4}$$

$$y_k^{(2)} = z_k^{(2)} \tag{5}$$

where $z_k^{(2)}$ is the input of the $k$th output neuron; $w_{jk}^{(2)}$ is the weight between the $j$th hidden neuron and the $k$th output neuron; $b_k^{(2)}$ is the bias of the $k$th output neuron; $y_k^{(2)}$ is the output of the $k$th output neuron.

The evaluation index, root mean square error (RMSE), of the SFNN is defined as

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\sum_{k=1}^{n_2}\left(y_k(t) - y_k^{(2)}\right)^2} \tag{6}$$

where $y_k(t)$ is the desired output of the $k$th output node for the $t$th sample, and $N$ is the total number of the samples.

### 2.2. Mutual Information (MI)

Entropy and MI are two basic conceptions in information theory. MI is generally used to measure statistical dependency between random variables and the entropy is used to represent information that contains in a given variable. The entropy of a discrete random variable $X = \{x_k | k = 1, \ldots, K\}$is defined as

$$H(X) = -\sum_{k=1}^{K} p(x_k)\log\ p(x_k) \tag{7}$$

where $p(.)$denotes the probability density function of $X$ and $K$ is the dimension of variable $X$. The joint entropy represents the information among multiple variables. For bivariate discrete random variables $(X, Y)$, the joint entropy is defined as

$$H(X,Y) = \sum_{x\in\chi}\sum_{y\in\gamma} p(x,y)\log\ p(x,y) \tag{8}$$

where $p(x,y) = p\{X = x, Y = y\}, x \in \chi, y \in \gamma$ is the joint distribution law of $(X, Y)$. $\chi$ and $\gamma$ are the value spaces of $X$ and $Y$ respectively.

The MI between two given discrete random variables $X$ and $Y$ is

$$I(X;Y) = \sum_{x\in\chi}\sum_{y\in\gamma} p(x,y)\log\frac{p(x,y)}{p_X(x)p_Y(y)} \tag{9}$$

where $p_X(x)$ and $p_Y(y)$ are the marginal distribution law of $X$ and $Y$, respectively. The MI describes the shared information of $X$ and $Y$. The larger the MI is, the stronger correlation is suggested between the two variables. The MI can be calculated by the entropy as follows

$$I(X;Y) = H(X) + H(Y) - H(X,Y). \tag{10}$$

## 3. Mutual Information based weight initialization (MIWI)

In this section, we introduce the MI estimation of SFNN firstly and then describe the process of MIWI algorithm. The MIWI algorithm initializes weights according to the information distribution in input variables. The bounds of the value space of weights are calculated to guarantee all the initial neurons are active. The ultimate aims are as follows: to guarantee the initial weights to get close to the optimal point with a higher probability; to ensure the performance stability and avoid premature saturation.

### 3.1. Mutual Information estimation of SFNN

The common methods to calculate MI include copula entropy method [32], Kernel method [33], $k$-nearest neighbor method [34] and histogram method [35,36]. In this paper, we adopt histogram method to estimate MI because it is the fastest and simplest method among them.

Assume an input variable and an output variable of SFNN are $X$ and $Y$. Firstly, estimate the entropy of variable $X$. The histogram method divides the value space of $X$ into $K_X$ segments with equal step $\Delta x$. To increase the accuracy of entropy estimation, the $K_X$ can be calculated by the method proposed by Hacine-Gharbi [35]

$$K_X = round\left\{\frac{\varepsilon}{6} + \frac{2}{3\varepsilon} + \frac{1}{3}\right\}, \tag{11}$$

$$\varepsilon = \sqrt[3]{8 + 324N + 12\sqrt{36N + 729N^2}} \tag{12}$$

where $N$ denotes the total number of samples. Due to the variable $X$ is a discrete random variable, the entropy of $X$ can be expressed as

$$H(X) = -\sum_{k=1}^{K_X} p_k(x)\log\ p_k(x) \tag{13}$$

where $p_k(x)$ means the probability of a sample drop-in the $k$th segment. It can be estimated as

$$p_k(x) = \frac{n_k}{N} \tag{14}$$

where $n_k$ is the number of samples observed in the segment $k$. Then the entropy can be expressed as

$$H(X) = -\sum_{k=1}^{K_X}\frac{n_k}{N}\log\frac{n_k}{N} \tag{15}$$

The entropy of variable $Y$ can be estimated the same as variable $X$.

Secondly, estimate the joint entropy of variables $X$ and $Y$. The histogram method divides the $xy$-plane into $(Kx \times K_Y)$ cells with equal size$(\Delta x \times \Delta y)$. The joint entropy of $X$ and $Y$ can be estimated as

$$H(X,Y) = -\sum_{k_i=1}^{K_X}\sum_{k_j=1}^{K_Y} p_{k_ik_j}(x,y)\log\ p_{k_ik_j}(x,y) \tag{16}$$

where $p_{k_ik_j}(x,y)$means the probability of a sample drop-in cell ($k_i$, $k_j$). It can be estimated as

$$p_{k_ik_j}(x,y) = \frac{n_{k_ik_j}}{N} \tag{17}$$

where $n_{k_ik_j}$ is the number of samples observed in cell ($k_i$, $k_j$). Then the joint entropy is expressed as

$$H(X,Y) = -\sum_{k_i=1}^{K_X}\sum_{k_j=1}^{K_Y}\frac{n_{k_ik_j}}{N}\log\frac{n_{k_ik_j}}{N} \tag{18}$$

The MI can be calculated with the combination of (Eqs. (10), 15) and (18)

$$I(X;Y) = -\sum_{k_i=1}^{K_X}\frac{n_{k_i}}{N}\log\frac{n_{k_i}}{N} - \sum_{k_j=1}^{K_Y}\frac{n_{k_j}}{N}\log\frac{n_{k_j}}{N} + \sum_{k_i=1}^{K_X}\sum_{k_j=1}^{K_Y}\frac{n_{k_ik_j}}{N}\log\frac{n_{k_ik_j}}{N}. \tag{19}$$

### 3.2. Initialization of input weights and biases of hidden neurons

Input weights of hidden neurons connect input neurons and hidden neurons. Inspired by the research of Talaska [17], the global optimal point is the one that can reflect the relationship between input variables and output variables. We initialize the input weights of hidden neurons according to the distribution of useful information in the input variables. The useful information in input variables is measured by MI between input variables and output variables, and it is defined as

$$MI(i) = \sum_{j=1}^{n_2} I\left[x_i; y_j\right], i = 1, 2, \ldots, n_0 \tag{20}$$

where $x_i$ is the $i$th input variable and $y_j$ is the $j$th output variable, $n_0$ is the number of input neurons and $n_2$ is the number of output neurons. Rank the input variables according to the useful information from small to large, and then the first input neuron contains the least amount of useful information.

Since the sigmoid function is a monotone increasing function, the weights should be larger if the useful information in the connected input neuron is greater. Assume the range of the input weights and biases of hidden neurons is [-$m$, $m$] and the weights connected to the $i$th input neuron is $w_i = [w_{i1}, w_{i2}, \ldots, w_{in_1}]$. We divide the value space into $n_0$ disjoint fragments and arrange the fragments from small to large. Then the initial value of $w_i$ is set as random values in the $i$th fragment. The width of the $i$th fragment is calculated as follows

$$k(i) = \frac{2m \times MI(i)}{\sum_{j=1}^{n_0} MI(j)} \tag{21}$$

To ensure the initial neurons be active in the initial phase, the scope of the value space should be limited. The active function used in this paper is the conventional sigmoidal function

$$f(a) = \frac{1}{1+e^{(-a)}}. \tag{22}$$

In the case, the active region is generally the region where the derivative of the active function is greater than one-twentieth of the maximum derivative [18,19,21]. Then the active region is

$$-4.36 \le a \le 4.36 \tag{23}$$

The maximal input value of one hidden neuron produced by the $i$th input neuron is

$$\max(s_i) = -m + \sum_{j=1}^{i} k(j) \tag{24}$$

Then the maximal input value of one hidden neuron is

$$\max\left(z^{(1)}\right) = \sum_{i=1}^{n_0} \max(s_i) = -n_0 m + n_0 k(1) + (n_0 - 1)k(2) + \cdots + k(n_0) + m \tag{25}$$

The maximal value of $k(.)$ is

$$\max(k) = 2m \frac{\max(MI)}{\sum_{j=1}^{n_0} MI(j)} = 2mA \tag{26}$$

$$A = \frac{\max(MI)}{\sum_{j=1}^{n_0} MI(j)} \tag{27}$$

According to (Eqs. (25) and 26)

$$\begin{aligned}\max\left(z^{(1)}\right) &\le -n_0 m + 2n_0 mA + 2(n_0 - 1)mA + \cdots + 2mA + m \\ &= -n_0 m + n_0(n_0 + 1)mA + m\end{aligned} \tag{28}$$

If the right side of formula (28) is smaller than 4.36, the initial input of hidden neurons can be guaranteed to be in the active region. Then the range of $m$ can be obtained

$$m \le \frac{4.36}{n_0(n_0+1)A - n_0 + 1} \tag{29}$$

In some extreme conditions, the value calculated with formula (29) is too small, which influences the network performance. To avoid this problem, we set $m$ as the following formula

$$m = \begin{cases} 0.5 & m \le 0.5 \\ \frac{4.36}{n_0(n_0+1)A - n_0 + 1} & m > 0.5 \end{cases} \tag{30}$$

To ensure the diversity of hidden neurons, the biases are uniformly randomly distributed over the interval $[-m, m]$.

### 3.3. Initialization of input weights and biases of output neurons

The initialization of input weights of hidden neurons presented above reflects the information distribution in input variables. To guarantee the performance of the algorithm, the input weights and biases of output neurons also should be designed carefully [14,21].

In recent years, many researchers compared the performance of different weight initialization methods. Thimm [15] compared several random weight initialization algorithms and concluded that the weights in the range $[-0.77, 0.77]$ seem to give the best mean performance. Fernandez-Redonodo [37] compared the performance of seven weight initialization methods based on extensive experiments. The results showed that the method proposed by Shimodaira [38], based on geometrical considerations, was the best weight initialization method for BP algorithm, but it needs preprocessing. Yang [22] proposed a new standard for split-complex backpropagation algorithm (SCBP) and the new algorithm was better than other ten weights initialization algorithms. Recently, Adam [21] set the parameter of the method introduced by Wessels [14] to be different values for different network structure, and the improved method was used to initialize the input weights of output layer. All these methods are completely random in a fixed interval to initialize the input weights of hidden and output layers, so they are not suitable for our method.

Since information distribution is reflected by the initial input weights of hidden neurons, the value range of the initial input weights and biases of output neurons should be large enough to guarantee the initial point is close to the global optimal point. Through many experiments, we found the random weights that are uniformly distributed over the interval $[-1, 1]$ as introduced in Fahlman [39] are the most suitable initial input weights of output layer for our method.

### 3.4. Procedure of MIWI Algorithm

The MIWI method for SFNN initializes the input weights of hidden layer based on the distribution of useful information in input variables. The input weights of hidden layer are set within a range to guarantee all the hidden neurons are active in the initial stage. The biases are distributed in the value space randomly and uniformly to ensure the diversity of hidden neurons. The range of input weights and biases of output neurons is large enough to ensure the initial point is close to the global optimal point. The aim of this method is to make the SFNN have higher chance to converge to the optimal point stably and fast. The main procedure of the MIWI algorithm is shown in Table 2.

## 4. Simulations

To examine the effectiveness of MIWI, three kinds of problems are performed by the algorithm: two classification problems, two function approximation tasks and one practical problem for prediction of the effluent total phosphorus (ETP) in the wastewater treatment system. The results are compared with other well-known

initialization algorithms. Since Backpropagation (BP) algorithm is the most popular GB training algorithm, without loss of generality, we used BP algorithm to train the neural networks. All experiments were run with Matlab version 10.0b on a PC with Windows XP operating system, Pentium IV 1.8 GHz CPU and 504 MB SDRAM memory.

### 4.1. Data classification

In this experiment, two benchmark classification problems were chosen from UCI Machine Learning Repository, namely, Iris and Diabetes. This kind of experiments is motivated by the importance and universality of classification problems. The basic features of the two datasets are shown in Table 3. The datasets of the two problems were divided into two sets: 75% were randomly selected for the training set and the remaining 25% were for the test set. The number of the data in each set is shown in Table 3.

The original datasets were normalized to $[-1,1]$. For each classification problem neural network architecture was designed based on experiments. Experiments with different architectures were performed and the best one was selected. BP algorithm was used for training the networks. The architectures of the networks and training parameters are displayed in Table 4.

The performance of MIWI is compared with other weights initialization algorithms: the method proposed by Thimm [15], the method proposed by Yam [19], the linear interval tolerance approach (LIT-Approach) proposed by Adam [21] and the method verified by Fahlman [39].

The comparison of experimental results is shown in Table 5. The following parameters are used to measure the performance: time for initialization, mean training time, mean precision and the variance of test results of ten consecutive experiments. From Table 5, the following comments can be made:

1) The mean precision of the MIWI algorithm is that the highest and the variance of test results are the lowest in the experiments.
2) Although initialization time of the MIWI is longer than other algorithms, but is much smaller compared with the training time. The mean training time of the MIWI is less than other methods obviously.

**Table 2**
MIWI algorithm.

1. Calculate the MI between each input variable and output variable with Eq. (19).
2. Calculate the total MI of each input variable with Eq. (20).
3. Sort the input variables according to the useful information in ascending order.
4. Use Eq. (30) to calculate the value range $[-m, m]$ of input weights and biases of hidden layer.
5. Use Eq. (21) to calculate the width of the $n_0$ value spaces. Then the value space of the weights connected to the $i$th input neuron is $\left[-m+\sum_{j=1}^{i-1} k(j), -m+\sum_{j=1}^{i} k(j)\right]$.
6. The biases of hidden neurons are uniformly randomly distributed over the interval $[-m, m]$ to ensure the diversity of hidden neurons.
7. Input weights and biases of output neurons are uniformly randomly distributed over the interval $[-1, 1]$.

**Table 3**
Characteristics of two UCI benchmarks.

| Dataset | Patterns | Training patterns | Test patterns | Input variables | Classes |
|---|---|---|---|---|---|
| Iris | 150 | 120 | 30 | 4 | 3 |
| Diabetes | 768 | 590 | 178 | 8 | 2 |

**Table 4**
Architectures of networks and training parameters used for classification experiments.

| Benchmark | Network architecture | Activation function[a] | Learning rate[b] | Max epochs[c] | Desired accuracy[c] |
|---|---|---|---|---|---|
| Iris | 4-8-3 | logsig | 0.1 | 30 | 100% |
| Diabetes | 8-8-2 | logsig | 0.1 | 200 | 80% |

[a] logsig denotes the logistic sigmoid function.
[b] Learning rate is the initial learning rate used for training.
[c] Training stops when Max epochs is reached or the accuracy is higher than the Desired accuracy.

**Table 5**
Performance comparison of the different algorithms for classification (average value of 10 consecutive experiments).

| Data set | Algorithms | initialization time(s)[a] | Mean training time(s) | Mean precision(%)[b] | Variances[c] |
|---|---|---|---|---|---|
| Iris | MIWI | 0.0188 | 0.9219 | **99.001** | **2.5874** |
| | Thimm | 0.0000 | 0.9844 | 96.3340 | 10.9931 |
| | Yam | 0.0000 | 0.9570 | 95.6660 | 12.4775 |
| | Adam | 0.0156 | 0.9375 | 97.3330 | 9.3906 |
| | Fahlman | 0.0000 | 0.9531 | 96.3340 | 6.0573 |
| Diabetes | MIWI | 0.0196 | 7.2953 | **78.6530** | **3.5144** |
| | Thimm | 0.0000 | 8.5542 | 77.4160 | 9.1187 |
| | Yam | 0.0000 | 10.5625 | 73.9320 | 12.0657 |
| | Adam | 0.0156 | 8.2750 | 76.1810 | 10.3890 |
| | Fahlman | 0.0000 | 10.8594 | 75.0570 | 6.9498 |

[a] 0.0000 means the time for initialization is less than $10^{-4}$.
[b] precision means the test accuracy.
[c] Variances are the variance of test results of ten consecutive experiments.

### 4.2. Function approximation

Non-linear function approximation experiments are conducted to demonstrate the effectiveness of a weights initialization algorithm for nonlinear approximation. The benchmark functions used in this paper are defined as follows.

Function 1. The non-linear function for this benchmark is used by Mittal [40] to verify his weights initialization method.

$$y = \exp(x_1 \ \sin(\pi x_2))$$

Function2. The function used here is a non-linear function considered by Adam [21].

$$y = 0.5 \ \sin(\pi x_1^2) \sin(2\pi x_2)$$

Seven hundred groups of $[x_1, x_2]$ are randomly selected in the interval $[-1, 1] \times [-1, 1]$ for functions 1 and 2. Among them five hundred groups of data are used as training set and the rest are testing set. The network structure used in function 1 is (2-15-1) and in function 2 is (2-21-1). Both the networks are trained using the BP algorithm.

The learning rate for all the experiments are 0.1. The training will stop if 10,000 epochs is reached. All the experiments are trained for the same epochs to compare their performance. The detailed results are shown in Table 6. The following parameters were used to measure the performance: time for initialization, mean training time, mean RMSE and the variance of test RMSE of ten consecutive experiments. The compared algorithms are: the Thimm [15], the Yam [19], the Adam [21], the Yang [22] and the Fahlman [39].

From Table 6, it can be seen that the MIWI algorithm achieve the highest approximation accuracy and the variance of test RMSE is the smallest. For functions 1 and 2, the training time of different algorithms are nearly the same, which is because the training

**Table 6**
Performance comparison of the different algorithms for function approximation (average value of 10 consecutive experiments).

| Benchmarks | Algorithms | initialization time(s)[a] | Mean training time(s) | Mean RMSE | Variances[b] |
|---|---|---|---|---|---|
| Function 1 | MIWI | 0.0234 | 139.8594 | **0.0199** | $\mathbf{3.5494 \times 10^{-6}}$ |
| | Thimm | 0.0000 | 131.3125 | 0.0269 | $4.0036 \times 10^{-5}$ |
| | Yam | 0.0000 | 133.3125 | 0.0225 | $5.7154 \times 10^{-5}$ |
| | Adam | 0.0156 | 131.3570 | 0.0221 | $2.4067 \times 10^{-5}$ |
| | Yang | 0.0000 | 144.5983 | 0.0220 | $2.5011 \times 10^{-5}$ |
| | Fahlman | 0.0000 | 132.4375 | 0.0258 | $1.4490 \times 10^{-4}$ |
| Function 2 | MIWI | 0.0625 | 140.3438 | **0.0041** | $\mathbf{4.6678 \times 10^{-7}}$ |
| | Thimm | 0.0000 | 141.0469 | 0.0071 | $2.9849 \times 10^{-6}$ |
| | Yam | 0.0000 | 138.7031 | 0.0047 | $1.9196 \times 10^{-6}$ |
| | Adam | 0.0313 | 142.1094 | 0.0050 | $1.2579 \times 10^{-6}$ |
| | Yang | 0.0000 | 139.2800 | 0.0078 | $1.4884 \times 10^{-6}$ |
| | Fahlman | 0.0000 | 143.4063 | 0.0071 | $4.8604 \times 10^{-6}$ |

[a] 0.0000 means the time for initialization is less than $10^{-4}$.
[b] Variances are the variance of test results of ten consecutive experiments.

epochs of all the experiments are the same. The initialization time of MIWI is the longest compared with other algorithms but is much smaller than training time.

### 4.3. *Effluent total phosphorus (ETP) prediction*

In recent decades, wastewater treatment gets more and more attention from the country and the local governments. Ensuring that wastewater discharge is standard has important significance for environmental governance. ETP is one of the main indicators for wastewater treatment which is time-consuming to measure. So predicting ETP with other indicators, which are easy to measure, is a simple and fast method [41–43]. To test the effectiveness of MIWI, we applied our method in predicting ETP.

In this section, we developed an ETP prediction model based on SFNN. In the experiment, the most important and easy-to-measured water quality parameters were selected: temperature (T), oxidation reduction potential (ORP), dissolved oxygen concentration (DO), potential of hydrogen (pH), total soluble solid (TSS) and influent total phosphorus (ITP) as the input variables. The data used in the experiment was collected from a wastewater treatment factory in Beijing, China. The initial data set was pre-processed, and 233 groups of the total data were used for training and 110 groups for testing.

All the training data and testing data were normalized to $[-1,1]$. The network structure used here is (6-15-1). The activation function, network training algorithm, learning rate and training stop condition are the same as part 4.2.

Fig.2 shows the prediction results of neural network model using MIWI where the red line is the real value and the blue line is the output of SFNN. Fig.3 shows the RMSE values of different algorithms in the training process plotted using a logarithmic scale. We can see from Fig.3 that the convergence rates of all the algorithms are nearly the same at the beginning. So the MIWI can guarantee the convergence rate effectively. The comparison of experimental results is shown in Table 7. The following parameters were used to measure the performance: time for initialization, mean training time, mean RMSE and the variance of test RMSE of ten consecutive experiments. The compared algorithms are: the Thimm [15], the Yam [19], the Adam [21] and the Fahlman [39].

It is shown in Table 7 that the prediction model with MIWI achieves the lowest mean RMSE and variance. It means the proposed MIWI algorithm achieves the highest prediction accuracy. The initialization time of MIWI is the longest compared with other algorithms but is much smaller than training time. Due to all the experiments are trained 10000 steps, the training time of all the experiments are nearly the same.

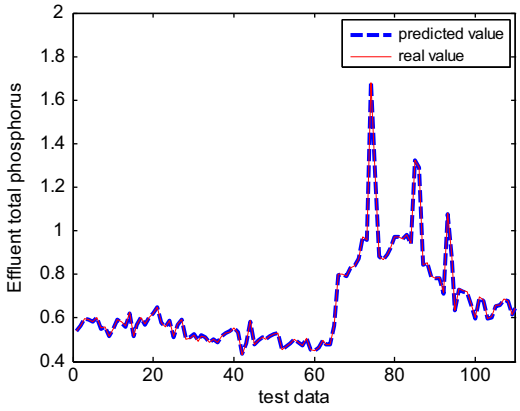


**Fig. 2.** The predicting results.

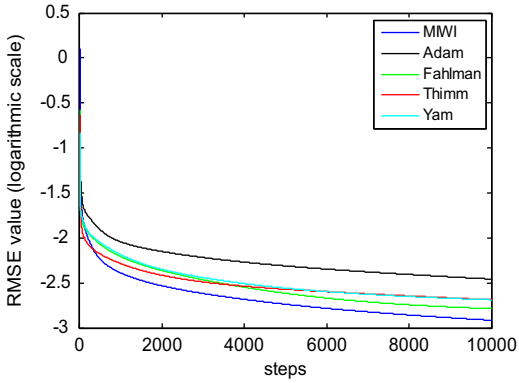


**Fig. 3.** The RMSE values in the training process with different algorithms.

**Table 7**
Performance comparison of the different algorithms for ETP predicting (average value of 10 consecutive experiments).

| | Algorithms | initialization time(s)[a] | Mean training time(s) | Mean RMSE | Variances[b] |
|---|---|---|---|---|---|
| ETP predicting | MIWI | 0.0116 | 81.4063 | **0.0025** | $\mathbf{1.0667 \times 10^{-7}}$ |
| | Thimm | 0.0000 | 81.3281 | 0.0028 | $1.6767 \times 10^{-7}$ |
| | Yam | 0.0000 | 81.7344 | 0.0034 | $3.1067 \times 10^{-7}$ |
| | Adam | 0.0089 | 81.9063 | 0.0034 | $7.3067 \times 10^{-6}$ |
| | Fahlman | 0.0000 | 81.5938 | 0.0033 | $1.1404 \times 10^{-6}$ |

[a] 0.0000 means the time for initialization is less than $10^{-4}$.
[b] Variances are the variance of test results of ten consecutive experiments.

## 5. Discussion

The aim of this paper is to present a stable weights initialization method that makes the initial weights get close to the global optimal point with a higher probability and avoids premature saturation. This section discusses the MIWI method based on its main idea and observations from experiments.

### 5.1. *Main idea*

The MIWI algorithm initializes weights according to the distribution of useful information in input variables. This mechanism makes the initial weights have high possibility to get close to the

global optimal point. Thus it can ensure the algorithm is stable and effective. The range of initial weights is calculated to ensure all the hidden neurons are active in the initial phase. Then the convergence rate can be guaranteed and the premature saturation can be avoided.

### 5.2. Accuracy of MIWI

Accuracy is an important index for neural network, which mainly depends on the structure and the parameters. Because the network structure with different weights initialization algorithm is fixed, the accuracy of the experiments mainly depends on the parameters. Since the initial distribution of weights is consistent with the information distribution in the input variables, the initial point is more likely to get close to the global optimal point. So MIWI can achieve good approximation accuracy and classification accuracy. Tables 5–7 indicate that MIWI achieves better accuracy than other algorithms.

### 5.3. Stability of MIWI

Stability is another important index, which has a strong impact on the effectiveness of an algorithm. All the existing random weights initialization methods select all the weights randomly in a value space. The proposed MIWI algorithm guides the initial weight distribution through information distribution. Therefore, the performance can be more stable than other weights initialization method. Form Tables 5–7 we can see that the variance of the results of MIWI is much less than other algorithms, supporting that the MIWI algorithm is more stable.

### 5.4. Initialization time

The results of experiments show that the MIWI algorithm spends more initialization time. That is because most of the initialization time of MIWI is used to calculate the MI between input variables and output variables, while other algorithms [15,19,39] do not need any calculation before initializing weights. The calculating amount of Adam [21] is also less than the MIWI. But the initialization time of MIWI is shorter than 0.05 s. Compared with the training time, the influence of the initialization time can be neglected.

### 5.5. Convergence rate

The lower and the upper bounds of the initial weights are calculated by MIWI to guarantee all the hidden neurons are active in the initial phase. So the convergence rate can be guaranteed and the premature saturation can be avoided. It can be seen form Fig.3 that the convergence rates of all the algorithms are nearly the same at the beginning. Also from the whole process, the MIWI converges faster than other algorithms.

## 6. Conclusion

This paper presents a novel method that initializes weights of SFNN based on the MI between input variables and output variables. The effectiveness of the proposed MIWI algorithm has been proved when applied to the problems of classification, function approximation and ETP prediction. The simulation results show that the MIWI algorithm has better performance in the aspects of stability and accuracy than other weights initialization algorithms. The advantages of the MIWI algorithm can be concluded as follows:

(1) The algorithm measures the distribution of useful information in input variables by the MI method. The initial distribution of weights is consistent with the information distribution in the input variables. This mechanism ensures the initial point has more opportunity to get close to the global optimal point. The accuracy and stability of the algorithm can be guaranteed.
(2) The range of the value space of initial weights is calculated to ensure the hidden neurons inputs are within the active region of sigmoid function at initial phase. Then the convergence rate can be guaranteed and the premature saturation can be avoided.
(3) The effectiveness of MIWI has been demonstrated by three kinds of experiments. But the algorithm still has some places need to be improved. The input weights of output layer are random selected in the interval $[-1, 1]$, a more suitable selected mechanism can further improve the training speed of the algorithm.

## Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grant 61533002, National Science Fund for Distinguished Young Scholars under Grant 61225016, China Postdoctoral Science Foundation under Grant 2015M570911, the ChaoYang District Postdoctoral Research Foundation under Grant 2015ZZ-6, and the Basic Research Foundation Project of Beijing University of Technology under Grant 002000514315501. The authors would like to thank the teachers in Beijing Key Laboratory of Computational Intelligence and Intelligent System who often give us valuable advice.

## References

[1] S. Timotheou, A novel weight initialization method for the random neural network, Neurocomputing 73 (2009) 160–168.
[2] Y.Q. Luo, Z.D. Wang, G.L. Wei, F.E. Alsaadi, T. Hayat, State estimation for a class of artificial neural networks with stochastically corrupted measurements under Round-Robin protocol, Neural Netw. 77 (2016) 70–79.
[3] D. Erdogmus, O.F. Romero, J.C. Principe, A.A. Betanzos, E. Castillo, Linear-Least-Squares initialization of multilayer perceptrons through backpropagation of the desired response, IEEE Trans. Neural Netw. 16 (2) (2005) 325–337.
[4] M.B. Nasr, M. Chtourou, A self-organizing map-based initialization for hybrid training of feedforward neural networks, Appl. Soft Comput. 11 (2011) 4458–4464.
[5] Y.K. Kim, J. B. Ra, Weight value initialization for improving training speed in the backpropagation network, in: Proceedings of the 1991 IEEE International Joint Conference on Neural Networks, IEEE, 1991, pp. 2396–2401.
[6] Y.F. Yam, T.W.S. Chow, Determining initial weights of feedforward neural networks based on least squares method, Neural Process. Lett. 2 (2) (1995) 13–17.
[7] F. Ros, M. Pintore, A. Deman, J.R. Chretien, Automatical initialization of RBF neural networks, Chemom. Intell. Lab. Syst. 87 (2007) 26–32.
[8] Z. Man, K. Lee, D. Wang, Z. Cao, C. Miao, A new robust training algorithm for a class of single-hidden layer feedforward neural networks, Neurocomputing 74 (2011) 2491–2501.
[9] Y.F. Yam, T.W.S. Chow, C.T. Leung, A new method in determining initial weights of feedforward neural networks for training enhancement, Neurocomputing 16 (1997) 23–32.
[10] D. Erdogmus, O. F. Romero, J. C. Principe, A. A. Betanzos, E. Castillo, R. Jenssen, Accurate initialization of neural network weights by backpropagation of the desired response, in: Proceedings of the International Joint Conference on Neural Networks, IEEE, vol. 3, 2003, pp. 2005–2010.
[11] Y. Liu, Y. W. Chen, Weight initialization of feedforward neural networks by means of partial least squares, in: Proceedings of the 2006 International Conference on Machine Learning and Cybernetics, IEEE, 2006, pp. 3119-3122.
[12] G. Napolitano, F. Serinaldi, L. See, Impact of EMD decomposition and random initialisation of weights in ANN hindcasting of daily stream flow series: an empirical examination, J. Hydrol. 406 (2011) 199–214.
[13] Q. Song, Robust initialization of a jordan network with recurrent constrained learning, IEEE Trans. Neural Netw. 22 (12) (2011) 2460–2473.
[14] L.F.A. Wessels, E. Barnard, Avoiding false local initialization of minima by proper connections, IEEE Trans. Neural Netw. 3 (6) (1992) 899–905.

[15] G. Thimm, E. Fiesler, High-order and multilayer perceptron initialization, IEEE Trans. Neural Netw. 8 (2) (1997) 349–359.
[16] S. Osowski, New approach to selection of initial values of weights in neural function approximation, Electron. Lett. 29 (3) (1993) 313–315.
[17] T. Talaska, M. Kolasa, R. Dlugosz, P.A. Farine, An efficient initialization mechanism of neurons for winner takes all neural network implemented in the CMOS technology, Appl. Math. Comput. 267 (2015) 119–138.
[18] J.Y.F. Yam, T.W.S. Chow, A weight initialization method for improving training speed in feedforward neural network, Neurocomputing 30 (1–4) (2000) 219–232.
[19] J.Y.F. Yam, T.W.S. Chow, Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients, IEEE Trans. Neural Netw. 12 (2) (2001) 430–434.
[20] G.P. Drago, S. Ridella, Statistically controlled activation weight initialization (SCAWI), IEEE Trans. Neural Netw. 3 (4) (1992) 627–631.
[21] S.P. Adam, D.A. Karras, G.D. Magoulas, M.N. Vrahatis, Solving the linear interval tolerance problem for weight initialization of neural networks, Neural Netw. 54 (2014) 17–37.
[22] S.S. Yang, S. Siu, C.L. Ho, Analysis of the initial values in split-complex back-propagation algorithm, IEEE Trans. Neural Netw. 19 (9) (2008) 1564–1573.
[23] J.F. Qiao, H.G. Han, Identification and modeling of nonlinear dynamical systems using a novel self-organizing RBF-based approach, Automatica 48 (8) (2012) 1729–1734.
[24] Z.Z. Zhang, Q.L. Chen, J.F. Qiao, A merging and splitting algorithm based on mutual information for design neural networks, in: Proceedings of the 2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), (IEEE, 2010), pp. 1268–1272.
[25] H.C. Peng, F.H. Long, C. Ding, Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1226–1238.
[26] C. Chen, X.F. Yan, Optimization of a multilayer neural network by using minimal redundancy maximal relevance-partial mutual information clustering with least square regression, IEEE Trans. Neural Netw. Learn. Syst. 26 (6) (2015) 1177–1187.
[27] M. Rana, I. Koprinska, Forecasting electricity load with advanced wavelet neural networks, Neurocomputing 182 (2016) 118–132.
[28] A.J. May, G.C. Dandy, H.R. maier, J.B. Nixon, Application of partial mutual information variable selection to ANN forecasting of water quality in water distribution systems, Environ. Model. Softw. 23 (10–11) (2008) 1289–1299.
[29] P. Henríquez, J.B. Alonso, M.A. Ferrer, C.M. Travieso, J.R. Orozco-Arroyave, Nonlinear dynamics characterization of emotional speech, Neurocomputing 132 (2014) 126–135.
[30] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (5) (1989) 359–366.
[31] K.L. Funahashi, On the approximate realization of continuous mappings by neural networks, Neural Netw. 2 (3) (1989) 183–192.
[32] M. Han, W.J. Ren, Global mutual information-based feature selection approach using single-objective and multi-objective optimization, Neurocomputing 168 (2015) 47–54.
[33] N. Kwak, C.H. Choi, Input feature selection by mutual information based on Parzen window, IEEE Trans. Pattern Anal. Mach. Intell. 24 (12) (2002) 1667–1671.
[34] M. Han, W.J. Ren, X.X. Liu, Joint mutual information-based input variable selection for multivariate time series modeling, Eng. Appl. Artif. Intell. 37 (2015) 250–257.
[35] A. Hacine-Gharbi, P. Ravier, R. Harba, T. Mohamadi, Low bias histogram-based estimation of mutual information for feature selection, Pattern Recognit. Lett. 33 (10) (2012) 1302–1308.
[36] M. Wei, T.W.S. Chow, R.H.M. Chan, Heterogeneous feature subset selection using mutual information-based feature transformation, Neurocomputing 168 (2015) 706–718.
[37] M. Fernández-Redondo, C. Hernández-Espinosa, Weight initialization methods for multilayer feedforward, in: Proceedings of the European symposium on artificial neural networks (ESANN), 2001, pp. 119–124.
[38] H. Shimodaira, A weight value initialization method for improved learning performance of the back propagation algorithm in neural networks, in: Proceedings of the 1994 the 6th International Conference on Tools with Artificial Intelligence, IEEE, 1994, pp. 672-675.
[39] S.E. Fahlman, An Empirical Study of Learning Speed in Back-propagation Networks, Technical Report CMU-CS-88-162, School of Computer Science, Carnegie Mellon University Pittsburgh, PA, 1988.
[40] A. Mittal, P. Chandra, A. P. Singh, A statistically resilient method of weight initialization for SFANN, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2015, pp. 1371-1376.
[41] S.H. Hong, Monitoring of sequencing batch reactor for nitrogen and phosphorus removal using neural networks, Biochem. Eng. J. 35 (2007) 365–370.
[42] D. Aguado, J. Ribes, T. Montoya, J. Ferrer, A. Seco, A methodology for sequencing batch reactor identification with artificial neural networks: a case study, Comput. Chem. Eng. 33 (2009) 465–472.
[43] M. Bagheri, S.A. Mirbagheri, M. Ehteshami, Z. Bagheri, Modeling of a sequencing batch reactor treatingmunicipal wastewater using multi-layer perceptronand radial basis function artificial neural networks, Process. Saf. Environ. Prot. 93 (2015) 111–123.
[44] W.J. Puma-Villanueva, E.P. dosSantos, F.J.V. Zuben, A constructive algorithm to synthesize arbitrarily connected feedforward neural networks, Neurocomputing 75 (2012) 14–32.

**Junfei Qiao** received the B.E. and M.E. degrees in control engineering from the Liaoning Technical University, Fuxin, China, in 1992 and 1995, respectively, and the Ph.D. degree from the Northeast University, Shenyang, China, in1998. From 1998 to 2000, he was a Postdoctoral Fellow with the School of Automatics, Tianjin University, Tianjin, China. He is currently a Professor with the Beijing University of Technology, Beijing, China. His current research interests include neural networks, intelligent systems, self-adaptive/learning systems, and process control systems.

**Sanyi Li** received his bachelor degree in electric engineering from the Henan Polytechnic University, Jiaozuo, China, in 2010 and his master degree in control engineering from the Henan Polytechnic University, Jiaozuo, China, in 2013. He is currently working toward the Ph.D. degree in the Beijing University of Technology, Beijing, China. His current research interests include neural networks and intelligent systems.

**Wenjing Li** received her bachelor degree in Xi'an Jiaotong University in 2007, and doctoral degree in Institute of Automation, Chinese Academy of Sciences in 2013. She is now a lecturer in Beijing University of Technology. Her main research areas are cognitive neuroscience, (pattern recognition and intelligent system).