Selection of Minimum Training Data for Generalization and On-line Training by Multilayer Neural Networks

Kazuyuki HARA †* Kenji NAKAYAMA†‡
†Graduate School of Nat. Sci. & Tech., Kanazawa University
* Toyama Polytechnic College
‡Dep. of Elec. and Comp. Eng., Faculty of Engineering, Kanazawa University
2-40-20, Kodatsuno, Kanazawa 920, JAPAN
E-mail: nakayama@ec.t.kanazawa-u.ac.jp

ABSTRACT

A training data reduction method for a multilayer neural network (MLNN) is proposed in this paper. This method reduce the data by selecting the minimum number of training data that guarantee generality of the MLNN. For this purpose, two methods are used. One of them is a pairing method which selects the training data by finding the nearest data of the different classes. Data along the class boundary in data space can be selected. The other method is a training method, which used a semi-optimum MLNN in a training process. Since the MLNN classify data based on the distance from the network boundary, the selected data can locate close to the class boundary. So, if the semi-optimum MLNN did not select data from class boundary, pairing method can select them. The proposed methods can be applied to both off-line training and on-line training. The proposed method is also investigated through computer simulation.

1. Introduction

In the classification problems, a multilayer neural network (MLNN) trained by supervised learning algorithms are capable of extracting common features or rules of training data through a training process. This is a benefit of using the MLNN for the classification. However, the suitable architecture of the MLNN and a small numbers of training data are required. The error back-propagation(BP)[1] algorithm is a popular algorithm for solving the classification problems.

One of the main interests of the supervised learning algorithms is how to select the training data. A huge amount of the training data may guarantee generality of the MLNN. On the other hand, it will require a very long training time. Therefore, it is desirable to reduce the number of the training data while maintaining generalization. Cachin [2] proposed the error-dependent repetition (EDR). Presentation probability of the training data is proportional to the MLNN output error. However, the entire data are used in the training process.

In this paper, we propose a method to select the efficient training data, with which generalization is guaranteed. The selected data can locate around the boundary between classes. This method can be applied to reduce in data memory and computations of off-line training, where a sufficient number of training data can be obtained in advance. Furthermore, it will be useful for an on-line training, where all training data can not obtained at the begining, rather they are gradually increased.

Efficiency of the proposed method is investigated through computer simulations. The BP algorithm is used to train the MLNN. Two kinds of problems are employed as examples.

2. Multilauer Neural Network

In this paper, a two-layer MLNN is used to classify the data. N samples of a piece of data $\boldsymbol{x} = \{x(i), i = 1 \sim N\}$ is applied to the input layer. The ith input unit receives $x(i)$. The connection weight from the ith input to the jth hidden unit is denoted w_{ij}. The input potential net_j and the output y_j of the jth hidden unit are given by

$$net_j = \sum_{i=1}^{N} w_{ij} x(i) + \theta_j \qquad (1a)$$

$$y_j = f_H(net_j), j = 1 \sim J \tag{1b}$$

$$f_H(net_j) = \frac{1 - e^{-net_j}}{1 + e^{-net_j}} \tag{1c}$$

where, $f_H(\cdot)$ is an activation function in the hidden layer and θ_j is a bias. The input potential net_k and the output y_k of the kth output unit are given by

$$net_k = \sum_{j=1}^{J} w_{jk} y_j + \theta_k \tag{2a}$$

$$y_k = f_O(net_k), k = 1 \sim P \tag{2b}$$

$$f_O(net_k) = \frac{1}{1 + e^{-net_k}} \tag{2c}$$

where $f_O(\cdot)$ is an activation function in the output layer.

The number of output units is equal to that of the classes. The MLNN is trained so that a single output unit responds to one of the classes.

3. Input Data and Unit Output

Input of the jth hidden unit is expressed by Eq.(1a). The input space can be separated into two regions by a line formed by $net_j = 0$ at the input of the hidden unit. A distance between this line and the input data is given by

$$d_j = \frac{\left|\sum_{n=1}^{N} w_{ij} x(i) + \theta_j\right|}{||\boldsymbol{w}_j||} = \frac{|net_j|}{||\boldsymbol{w}_j||}, \tag{3a}$$

$$\boldsymbol{w}_j = \{w_{ij}, i = 1 \sim N\}. \tag{3b}$$

$||\boldsymbol{w}_j||$ is an L_2 norm of the weight vector $\boldsymbol{w}_j$. Then the input potential net_j is proportional to the distance d_j. The activation function Eq.(1c) is a continuous monotonically increasing function, then the hidden unit output y_j is also continuous monotonically increasing with respect to the distance d_j. However, y_j is not a linear function of the distance.

The output of the output unit y_k is separated by the regions of $y_k > 0.5$ and $y_k < 0.5$. The input potential $net_k = 0$ provides a decision boundary. This is called a network boundary in this paper. The class boundary means the boundary of the input data classes. If the training converges, the network boundary will agree with the class boundary. Then a distance from the class boundary to a data is related to $|y_k - 0.5|$. In this case, the input potential of the output unit net_k is also related to the distance.

In conclusion, $|y_k - 0.5|$ and $|net_k|$ are continuous functions with respect to the distance between the data boundary and the input data.

4. Pairing Method for Data Selection

The proposed data selection method combines a training process and a pairing method. In this section, a pairing method is first described.

In this paper, two classes $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ are taken into account for convenience. However, the proposed method can be applied to more than two classes.

In the pairing process, the nearest data of different classes evaluated using the Euclidian distance are selected. Let $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ be sets of two data classes, and $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ be elements of them. $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are paired with each other through the following steps.

Step 1: Select $\boldsymbol{x}_1$ (or $\boldsymbol{x}_2$) from $\boldsymbol{X}_1$ (or $\boldsymbol{X}_2$) randomly.

Step 2: Select $\boldsymbol{x}_2^p$ (or $\boldsymbol{x}_1^p$) from $\boldsymbol{X}_2$ (or $\boldsymbol{X}_1$), which has the shortest distance to the $\boldsymbol{x}_1$ (or $\boldsymbol{x}_2$), selected in Step 1.

Step 3: Select $\boldsymbol{x}_1^p$ (or $\boldsymbol{x}_2^p$) from $\boldsymbol{X}_1$ (or $\boldsymbol{X}_2$), which has the shortest distance to $\boldsymbol{x}_2^p$ (or $\boldsymbol{x}_1^p$), selected in Step 2.

When all data are selected from $\boldsymbol{X}_1$ (or $\boldsymbol{X}_2$) in Step 1, the pairing process is completed. Otherwise, return to Step 1, and repeat the above process. In this process, the same data will not be selected. Finally, the data x_1^p and x_2^p, selected based on the distance, are included in the reduced data set.

If the class boundaries in the data space are based on the distance, the data located close to the boundary can be detected by this method.

5. Data Selection Method Using Training and Pairing

5.1. Data Selection Algorithm

This method combines the training and the pairing as follows:

Step 1: Some number of the training data are randomly selected from $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$. Let the sets of the selected data be $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$.

Step 2: Train the MLNN using the data in $\boldsymbol{X}_1^r$ and $\boldsymbol{X}_2^r$.

Step 3: Select the data, with which the network output errors have relatively large error. Let these data be $\boldsymbol{x}_1^e$ and $\boldsymbol{x}_2^e$.

Step 4: Select the data x_1^p and x_2^p from X_1^r and X_2^r, which have the shortest distance to x_2^e and x_1^e, respectively.

Step 5: Select the data x_1^{pe} and x_2^{pe} from X_1^p and X_2^p, which have the shortest distance to x_2^p and x_1^p, respectively.

A set of x_1^p, x_2^p and x_1^{pe}, x_2^{pe} will be used in the next training process. Replace the data in X_1^r and X_2^r by the new training data, and return to Step 2.

When new data are provided, they are included in X_1^r and X_2^r . The remaining data of X_1 and X_2 can be also used for this purpose. If the number of the new data is large, some number of the data may be selected, and are included in X_1^r and X_2^r . After that, return to Step 2.

The data selected in Step 3 satisfy

$$X_1^e = \{x_1 | y(x_1) < a_+, t = 1\} \tag{4a}$$
$$X_2^e = \{x_2 | y(x_2) > a_-, t = 0\} \tag{4b}$$

where $y(\cdot)$ express the output, t is the target, and a_+ and a_- express some levels, for instance 0.7 and 0.3, respectively.

5.2. Data Distribution by Training Method

Purpose of the training in Step 2 is to find the data, which locate close to the class boundary, with less computations. Therefore, the training is stoped at the middle stage in the training process using some criterion. In subsection 7.2.2, this criterion of an off-line training is described. Even though the training is not completely converged, the data, which locate close to the class boundary can be detected using the output error. The details are described in the following.

For convenience, a two-dimensional pattern classification given by Fig.1 (a) is employed. It is assumed that the triangle network boundary shown in Fig.1 (b) is formed in Step 2. The data inside the triangle corresponds to Class 1, and the outside Class 2. In this case, the regions are further divided into A, B, C and D as shown in Fig.1 (b). This means that the data locate in B and D are exactly classified into Class 1 and Class 2, respectively. Furthermore, the data in A and C are miss-classified into Class 2 and Class 1, respectively.

Following the process in Step 3, the data in A and C will remain due to large output error by miss-classification. Further, the data locate close to the

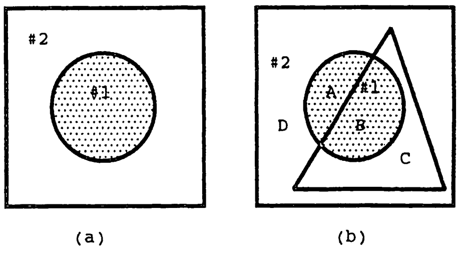


Fig. 1: (a) Class distribution, (b) Classification result by not well achieved network.

network boundary, in B and D are also detected due to relatively large errors. The error is highly related to the distance from the boundary. However, it is not always proportional to the distance. This will be discussed in Sec. 7.2.2. Therefore, the data, with which the output error is relatively large, locate near the network boundary, that is the triangle, at least.

However, the data, which locate close to the network boundary, do not cause large output error. Therefore if the data x_1^e and x_2^e are only selected, the efficient data, which locate close to the boundary, will be missed. Figure 3 shows an example, where the data locate in the shaded parts are only satisfy the conditions Eq.(4a) and Eq.(4b), and are detected.

For this reason, the pairing method is combined. The data in the different classes locate close to x_1^e and x_2^e can be found. They are denoted x_2^p and x_1^p as shown in Step 4, respectively.

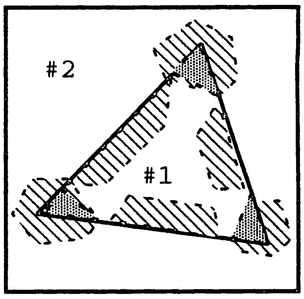


Fig. 2: Example of two-class classification.

6. Training Data Selection in Off-line and On-line Trainings

The proposed data selection method can be applied to both off-line training and on-line training [3]. In the off-line training, all data are given at the beginning of the training. If a large amount of training data is available, the data selection is needed to reduce the training time. In the on-line training, the training data are not given all together, but are given successively. Furthermore, they may change continuously. If the data successively received are all accumulated, then the number of the data will be extremely large. Therefore, in this application, the training data selection is very important.

7. Computer Simulation

Two-dimensional two-class classification is employed for computer simulations. The number of input unit N is 2, and the number of output unit K is 2. Then, The data is $\boldsymbol{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2\}$ and the input data is $\boldsymbol{x} = \{x(i), i = 1, 2\}$.

Figure 3 shows a concept of the problems. One of the classes is shown as shaded region, and the other is dotted region. White region between the classes shows a gap, so there is no overlap.

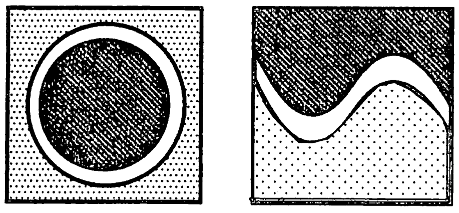


Fig. 3: Concept of problems. (a) Circle in square, (b) Sinusoidal in square.

For the training, learning-rate parameter η is 0.1, and momentum constant α is 0.8. These are decided by experience. Circle in square is called problem 1, and Sinusoidal in square is called problem 2 in following sections.

In problem 1, two classes are defined as follows:

$$\boldsymbol{X}_1 = \{\boldsymbol{x} \,|x(1)^2 + x(2)^2 \leq (r-\gamma)^2\} \tag{5a}$$
$$\boldsymbol{X}_2 = \{\boldsymbol{x} \,|x(1)^2 + x(2)^2 > (r+\gamma)^2\} \tag{5b}$$

here, r is the radius of the circle and is 0.39. γ is the width of the gap, and is 0.02.

In problem 2, two classes are defined as follows:

$$\boldsymbol{X}_1 = \{\boldsymbol{x} \,|A\sin(2\pi \cdot x(1)) \leq x(2) - \gamma\} \tag{6a}$$
$$\boldsymbol{X}_2 = \{\boldsymbol{x} \,|A\sin(2\pi \cdot x(1)) > x(2) + \gamma\} \tag{6b}$$

where, the A is the amplitude and is 0.22.

7.1. Computer Simulation Conditions

The number of data for each class is 1000. Six hidden units are used. Two hundreds of data for each class are selected randomly from 1000 data. These are used in following simulations.

7.2. Off-line training

7.2.1. Pairing Method

For off-line training, pairing and training methods are used. Figure 4 shows randomly selected data, and Fig.5shows the data found by pairing method. From Fig.5, the class boundary is formed by data properly. Sixty-five data are selected for each class.

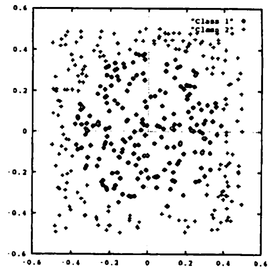


Fig. 4: Randomly selected data.

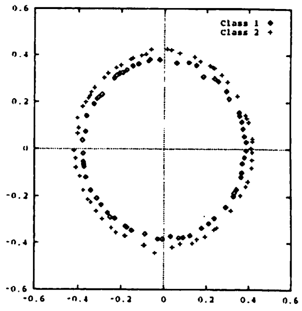


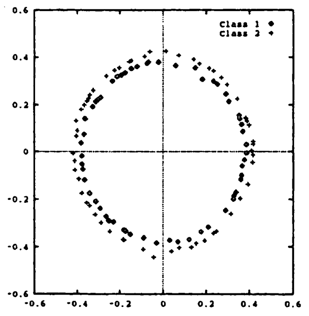
Fig. 5: Selected data by pairing method.

The MLNN is trained with selected data. The stopping criterion is 0.001 in the mean square error (MSE) at the output layer. Iteration of 23763 is needed for convergence.

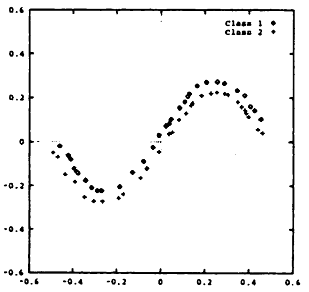
7.2.2. Training Method

The initial training is stopped at the MSE of ε <0.05. The thresholds, a_+ and a_-, are 1.0, and are equal to ε of 0.073. In problem 1, 207 of data are selected from Class 1, and 164 from Class 2. From Class 1 and Class 2, 116 and 150 of data are selected in problem 2.

Figure 6 shows the results. From these figures, the boundary is detected properly.



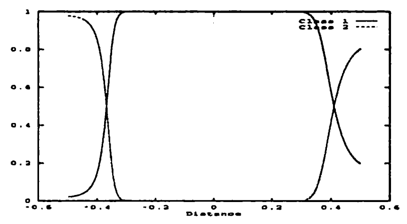
(a)Circle in square.



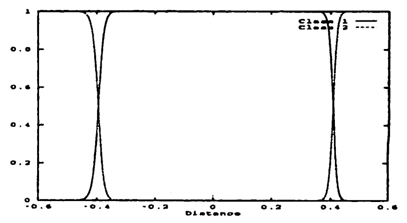
(b)Sinusoidal in square.
Fig. 6: Selected data

For stopping the training, four hundreds of validation data are used to have a consistent stopping criterion for conventional method and proposed one. The validation data are a subset of the entire data set. The network output error is calculated for validation data every iteration. The stopping criterion ε of the validation data is 0.001. Table 1 shows the results. In the table, computation of the conventional method is 1.0, and the computation of the proposed method is represented as a ratio of proposed method of conventional one. From this table, the computational complexity is reduced by this process.

Figure 7 shows relation among the distance and the output unit output. The MLNN in step 2 of Problem 1 is used. The input data of (a) is $x(2) = 0$ and (b) is $x(1) = x(2)$. In the figures, the horizontal axis is the distance from the origin of the data space to a data. The vertical axis is the output of the output unit to the input data of the horizontal axis. From the figures, (b) has much steeper slope than (a) near the class boundary, that is at $\pm$ 4.0 in the horizontal axis. Then data locate on (a) outputs different value from the data on (b) for the same distance as mentioned in subsection 5.2.



(a) Input data is $x(2) = 0$



(b) Input data is $x(1) = x(2)$

Fig. 7: Unit output and distance in data space

7.3. On-line training

The on-line training is simulated using partial data of the problems. Problem 1 is used in this simulation. Entire data $\boldsymbol{X}$ is separated into three sets as described below.

$$\begin{aligned} \boldsymbol{X}_{up} &= \{\boldsymbol{x} \,|x(2) \geq 0.167\} & \text{(7a)} \\ \boldsymbol{X}_{mid} &= \{\boldsymbol{x} \,|-0.167 < x(2) < 0.167\} & \text{(7b)} \\ \boldsymbol{X}_{down} &= \{\boldsymbol{x} \,|x(2) \leq -0.167\} & \text{(7c)} \end{aligned}$$

Each data subset includes 333 data.

$\boldsymbol{X}_{up}$ is used as the training data of Step 1. $\boldsymbol{X}_{mid}$ and $\boldsymbol{X}_{down}$ are used new training data in training of Step 2 of Sec.5. The stopping criterion ε in Step 1 is

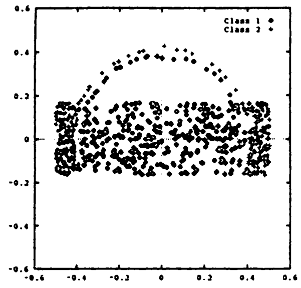
Table: 1: Comparison of computational complexity between conventional and proposed training.

	Prob. 1		Prob. 2	
	Conv.	Proposed	Conv.	Proposed
Init.	0	134	0	18
Epoch	2444	4394	89	390
N of data	2000	114	2000	62
Comp.	1.0	0.10	1.0	0.14

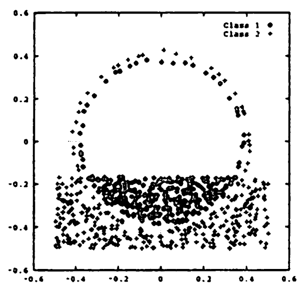
Init.: Epoch of initial training.
N of data: Number of data.
Conv.: Conventional method. Comp. Computation

0.05, and 0.01 for training convergence, respectively. The thresholds for all steps are 1.0.

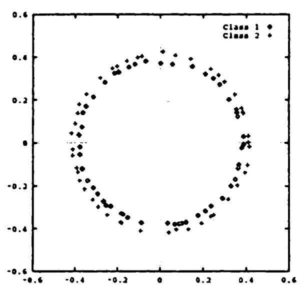
Figure 8 shows the result of on-line training using selected data. The training is converged and their percentage of correctly classified are 100 % for entire data set. The boundary is also detected properly.



(a) Training data of Step 1. Upper one third region is selected data and middle region is newly given data.



(b) Training data of step 2. Upper two third regions are selected data. MLNN is trained with data of (a). Rest of the region is newly given data.



(c) Selected data of Step 5. MLNN is trained with data of (b).

Fig. 8: Selected data: Problem 1.

8. Conclusion

The training data selection method used in the MLNN have been proposed. Pairing method using the Euclidean distance to find sets of the nearest data to the initially randomly selected data. The training method selects the data based on the network boundary of the MLNN. These methods are combined in this method. Validity of the training methods has been given, and it was confirmed that the training method never lost the data near the class boundary by using pairing method. Proposed methods have been applied to the applications. One of them is reducing the training of the off-line training, and the other is the on-line training. The training has been converged by paring method and training method. The computations to converge the training has been reduced. Training method is also applied to on-line training. In this case, data are selected from the partial data. The training has been converged. Therefore, proposed methods are supported by the simulation results.

References

[1] D.E.Rumelhart and J.L.McCelland et al., *Parallel Distributed Processing*, MIT Press, 1993.

[2] C. Cachin, "Pedagogical pattern selection strategies," *Neural Netwroks*, vol.7 No.1, pp.175–181, 1994.

[3] S.Haykin, *Neural Networks – A comprehensive foundation*, pp. 57–59, IEEE Press, 1994.

[4] K.Hara, K.Nakayama, "High resolution of multi-frequencies using miltilayer networks trained by back-propagation algorithm," *WCNN'93*, vol.IV, pp.675–678, 1993.

[5] T.M.Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Trans. Electric Computers*, vol. EC–14, pp. 326–334, 1965.

[6] K.Hara and K.Nakayama, "Comparison of activation functions in multilayer neural network for pattern classification," *Proc. ICNN'94*,Orland, Florida, vol. V, pp. 2997–3002, 1994.