A Training Method with Small Computation for Classification

メタデータ	言語: English 出版者: 公開日: 2017-10-03 キーワード (Ja): キーワード (En): 作成者: Hara, Kazuyuki, Nakayama, Kenji メールアドレス: 所属:
URL	http://hdl.handle.net/2297/6823

A Training Method with Small Computation for Classification

Kazuyuki Hara and Kenji Nakayama†
Tokyo Metropolitan College of Technology
†Faculty of Engineering, Kanazawa University
E-mail:hara@tokyo-tmct.ac.jp, nakayama@t.kanazawa-u.ac.jp

Abstract

A training data selection method for multi-class data is proposed. This method can be used for multilayer neural networks (MLNN). The MLNN can be applied to pattern classification, signal process, and other problems that can be considered as the classification problem. The proposed data selection algorithm selects the important data to achieve a good classification performance. However, the training using the selected data converges slowly, so we also propose an acceleration method. The proposed training method adds the randomly selected data to the boundary data. The validity of the proposed methods is confirmed through the computer simulation.

1. Introduction

Recently, Multilayer Neural Networks (MLNN) is widely used in pattern recognition, signal processing, image processing, and other many fields. In these fields, it is a main benefit of using the MLNN that extracts classification rules automatically in the training process. The network is required that the rule can classify both training data and the new data. This ability is called generalization.

If we have no strategy to select the efficient data to achieve generalization, we must use a huge number of data. In this case, the training becomes difficult. Moreover, the computational complexity becomes large. Therefore, data selection to achieve generalization is important task.

Many researchers proposed the data selection methods from different points. Active sampling[2],[3],[4] selects data by analyzing the stochastic specification of data. Stochastic sampling[5] selects data related to the output error. Dynamic sampling[6] selects data to avoid rank-deficiencies of connection weight matrix.

The MLNN divides the input space by hyper-planes formed by the connection weights. The network solves the classification problems by adapting the hyper-planes to the class boundaries. When the hyper-planes agree with the class boundaries, the network can achieve generalization. From [7, 8], it is shown that the important data to achieve generalization is located near the class boundaries. This is our course of data selection method proposed in this paper.

We propose a new data selection method that can be used for multi-class data. This method can apply to the data that the class regions are not overlapped. We also propose a novel training method trains using the class boundary data. By proposed method, the network can achieve generalization with small number of important data. We assume that the batch learning is employed. The validity of proposed method is verified by computer simulation.

2. Multilayer Neural Networks

2.1. Structure of Multilayer Neural Networks

The network used in this paper is a two-layer multilayer neural network that has the input layer, one hidden layer and the output layer. The number of the input units is the same as the dimension of the input data and the number of the output units is the same as the number of the classes.

The output of the MLNN is calculated as follows. We denote a input vector $\boldsymbol{x} = \{x_i, i = 1, \ldots, N\}$, then the input potential net_j and the output of the j-th hidden unit y_j are

0-7695-0619-4/00/$10.00 (C) 2000 IEEE

$$net_j = \sum_{i=1}^{N} w_{ji} x_i + \theta_j, \tag{1}$$

$$y_j = f_H(net_j) = \tanh(net_j). \tag{2}$$

Here, w_{ji} is the connection weight between the i-th input unit and the j-th hidden unit. $j = 0$ corresponds to the bias of the unit. $f_H(\cdot)$ is the sigmoid function.

The output y_j is translated to the output unit weighted by the connection weight w_{jk}. The input potential and its output of the output units are calculated at the same method as the hidden layer except for using uni-polar sigmoid function $f_O(\cdot)$ as shown in below.

$$y_k = f_O(net_k) = \frac{1}{1 + \exp(-net_k)} \tag{3}$$

To train the MLNN, Back-propagation algorithm is used. This algorithm updates the connection weights to decrease the mean square error (MSE) of the MLNN outputs. We denote all the data as $\boldsymbol{X} = \{\boldsymbol{x}_p, p = 1, \ldots P\}$, then MSE is calculated as

$$E = \frac{1}{P} \sum_{p=1}^{P} \frac{1}{K} \sum_{k=1}^{K} (t_{pk} - y_{pk})^2. \tag{4}$$

Here, the target for the k-th output unit of the p-th data is denoted as t_{pk}, and the k-th output for the pth data is denoted as y_{pk}, respectively.

2.2. Classification using Multilayer Neural Networks

The input potential of the k-th output unit is written as

$$net_q^O = \sum_{j=0}^{R} w_{kj} y_j. \tag{5}$$

Here, w_{kj} is the connection weight between the j-th hidden unit and the k-th output unit. We consider the classification problem, so the target t_k for the data x_k is "1" and the target for other class data is "0". Then the training data will be classified as following way.

$$\begin{cases} \boldsymbol{x} \in \boldsymbol{X}^k & \text{if } net_k^O > 0 \\ \boldsymbol{x} \notin \boldsymbol{X}^k & \text{if } net_k^O < 0 \end{cases} \tag{6}$$

Here, $\boldsymbol{X}^k$ is set of the data belongs to k-th class. From equation (6), $net_k^O = 0$ is the boundary of the class and from equation 3, the output of the unit is 0.5. Therefore, in the input space, the region of k-th output unit is larger than 0.5 can be considered as the k-th class region.

The pattern matching method is useful for the classification problem. It measures the distance from the data to the class template patterns and classifies into the class that measured the shortest distance from the data. The class region formed by the network is correspond to the template pattern of the pattern matching. So, after the training, the p-th data $\boldsymbol{x}_p$ will be classified into k-th class if the following equation is satisfied.

$$\boldsymbol{x}_p \in \boldsymbol{X}^k \text{if } y_k(\boldsymbol{x}) = \arg \max_{1 \le k' \le K} \{y_{k'}(\boldsymbol{x})\} \tag{7}$$

0-7695-0619-4/00/$10.00 (C) 2000 IEEE

3. Data Selection Method

3.1. Boundary Data and Classification Performance

Two classes' classification $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ are taken into account for convenience. However, the proposed method can be applied to more than two classes.

The data distribution and the class boundary are shown in figure 1. Data are drawn by circles and the boundary is drawn by the line. In the case of the hyper-plane formed by the MLNN is located as in the line in the figure, the hyper-plane can separate the data correctly. In the another word, if the hyper-plane can classify the data indicated by the doubled-circle and the one indicated by circled-dot, the other data can also be separated by the hyper-plane. They are called "boundary data" in this paper and they play important role in this paper.

3.2. Data Selection Method

In this section, the boundary data selection method is proposed. Basic idea of selecting the boundary data is finding the nearest data between the specified two classes. To measure the distance, the Euclidean distance is used. One of the data is denoted by $\boldsymbol{x}$, and set of data of class $c \in C$ is denoted by $\boldsymbol{X}^c$. C denotes all the classes.

Step 1: Select one of the class c.
Step 2: Select one of the data $\boldsymbol{x}_m^c \in \boldsymbol{X}^c$ randomly.
Step 3: Select one of the data $\boldsymbol{x}_S^{c'}, c' \neq c$ by

$$\boldsymbol{x}_S^{c'} = \arg \min_{\substack{c' \neq c \\ 1 \leq j \leq M}} \{d(\boldsymbol{x}_m^c, \boldsymbol{x}_j^{c'})\}$$

Step 4: Select one of the data $\boldsymbol{x}_{S'}^c$ by

$$\boldsymbol{x}_{S'}^c = \arg \min_{\substack{c \\ 1 \leq k \leq M}} \{d(\boldsymbol{x}_S^{c'}, \boldsymbol{x}_k^c)\}$$

Here, j, k, m are index of one of the data in the classes. Index S and S' show selected data. $d(\cdot)$ denotes the Euclidean distance. The data selection is ended if all the classes are selected in Step 1. The computational complexity of selecting the boundary data is $O(n^2)$.

$\boldsymbol{x}_S^{c'}$ and $\boldsymbol{x}_{S'}^c$ become a pair. We call them as "paired data" in the following sections. We also call the gather of the "paired data" as "boundary data".

4. Training Using Boundary Data

In this section, we assume that the number of hidden units is carefully designed so that the classification can be achieved by the given network. The boundary formed by the MLNN is called as "network boundary". The batch learning is used and the connection weights are updated after all the training data is presented.

4.1. Network Boundary and Data Distribution

In gradient decent algorithm, the connection weight is updated as shown in the next equation.

$$\Delta w_{ji} \propto \sum_{p=1}^{P} \frac{\partial E_p}{w_{ji}} \tag{8}$$

P is the number of training data and all the data is used to update the connection weight. From the equation (8), Δw_{ji} is updated toward the average of the E_p. Therefore, the distribution of the entire data is reflected for update of the connection weight.

0-7695-0619-4/00/$10.00 (C) 2000 IEEE

If we train the MLNN with the boundary data, the distribution of the entire data is not considered. In the figure **??**, we show an example of the network boundary displacement for two different data distributions. From equation (8), we assume that the network boundary will be placed at the center of the class distributions. In the figure 2, "o" shows the boundary data in class 1, and "x" show the boundary data in class 2. Black dots show the center of the distribution of each class.

When the boundary data are used as the training data, the network boundary will be placed as same as (a). Then, if the true data distributions are (b), good classification performance for new data will not be guaranteed.

4.2. Adding Randomly Selected Data to Boundary Data

One of the way to reflect the data distribution into the boundary data is adding randomly selected data.

If we added many data to the boundary data, the number of the training data is increased, and computational complexity of the training becomes rich. So, the number of randomly selected data must be small.

Training Using Boundary Data

The boundary data are selected by the proposed method in Section 3.2. If the network is trained by the boundary data, the training needs many iterations from following two reasons. The first reason is that the absolute value of the connection weights must be large to classify the similar data into the different class. This requires many iterations. The network must be sensitive to the change of the input. The second reason is that the totals of update of the connection weights to the boundary data will be canceled. Especially, when the network approaches to the class boundary, this phenomenon is remarkable [7].

We can solve this problem by adding the randomly selected data to the boundary data. Because, the update to randomly selected data is properly done and the training is preceded. To do so, one randomly selected data is need for a pair of the boundary data.

4.3. Computer Simulation of Adding Randomly Selected Data

One dimensional classification problem is used for simulation to show the validity of adding one randomly selected data to a pair of the boundary data.

The location of the boundary data in class 1 is $x_S^1 = -0.45$ and the one in class 2 is $x_S^2 = -0.5$. The location of the randomly selected data located near the boundary data in class 1 is $x_{R1}^1 = 0.0$ and the one in class 2 is $x_{R1}^2 = -1.0$. The location of the randomly selected data far from the boundary data in class 1 is $x_{R2}^1 = 5.0$ and the one in class 2 is $x_{R2}^2 = -5.0$. The simulation conditions are follows: (a) using only the boundary data (b) using the boundary data and x_{11}^R, (c) using the boundary data and x_{21}^R (d) using the boundary data and x_{12}^R, (e) using the boundary data and x_{22}^R (f) using the boundary data, x_{11}^R and x_{21}^R, (g) using the boundary data, x_{12}^R and x_{22}^R. The training is stopped when the EMS is below 0.01 or training iteration reaches to 10000. From the results, the numbers of iterations to converge are (a) 9807, (b) 3409, (c) 3269, (d) 3845, (e) 3600, (f) 2878, (g) 3567.

From the results, by adding the randomly selected data to the boundary data, the corrections of the connection weights are done in successfully. We then conclude that we need at least one randomly selected data to a pair of boundary data.

5. Computer simulation

To verify the validity of the data selection method and the training method is done by the computer simulations. We used two classification problems. "Circle in square" in Fig. 4 (a) that is a two-dimensional three classes' classification problem (problem 1), and "Iris classification" are four dimensional three classes' classification problem (problem 2).

5.1. Problem 1

The concept of the problem 1 is shown in Fig. 4 (a). The number of the data in a class is 1000. Between the class regions, there is the gap of 0.05. The gaps are not shown in Fig. 4(a). This is used to verify the validity of the boundary data selection method.

0-7695-0619-4/00/$10.00 (C) 2000 IEEE

The result of extraction of the boundary data is shown in Fig. 4 (b). From the figure, it is confirmed that the class boundary is properly extracted by the double pairing method. In the figure, $N_P = 315$ boundary data are selected.

5.2. Problem 2

Problem 2 is Iris classification [9]. The data consist with three classes. Input data is consist of four specifications of the sepal length, the sepal width, the petal length and the petal width. From the document of the database, one class is linearly separable from two classes, however the others are linearly non-separable. The number of the data in a class is 50. This database is provided by the UCI repository of machine learning database and domain theories. Fig. 5 shows class data distributions of sepal length and sepal width. There is some overlap between class 2 and 3, so the training can not be achieved 100% accuracy.

(1)Training Results

Seventeen boundary data are selected by the boundary data selection method. So, we added 9 randomly selected data, so totally, 26 training data are used. The training is stopped when the accuracy of 98% is achieved. The training is converged with 46 iterations and the accuracy for all the data is 98.7%.

(2) Comparison of Computational Complexity of Proposed Training and Conventional Training

The computational complexity of the proposed training and conventional training that uses all the data, are compared. The training is stopped when the accuracy is reached at 98%. From the computer simulation, the conventional training converged with 94 iteration. Then ratio of $(Proposedtraining/Conventionaltraining)$ is $(46 \times 26)/(94 \times 150) \sim 0.08$. Then the computational complexity is drastically reduced.

6. Conclusions

In this paper, assuming there is no-overlap between the class regions and off-line training, we have proposed the data selection method and the training that guarantees high classification performance. The proposed method has achieved high accuracy by train the network with the boundary data and the random data. The number of these data is sufficiently small.

References

[1] M. Plutowski and H. White, *Selection concise training sets from clean data*, IEEE Trans. Neural Networks, vol. 4, no. 2, pp. 305-318, 1993.

[2] J. Hwang, J. J. Choi, at el., *Query learning based on boundary search and gradient computation of trained multilayer perceptrons*, Proc. IJCNN'90, pp. 57-62, San Diego, 1990.

[3] E. B. Baum, *Neural net algorithm that learns in polynomial time for examples and queries*, IEEE Trans. Neural Networks, vol. 2, no. 1, pp. 5-19, 1991.

[4] R. Battiti, *Using mutual information for selection features in supervised neural net learning*, IEEE Trans. Neural Networks, vol. 5, no. 4, pp. 537-550, 1994.

[5] C. Cachin, *Pedagogical pattern selection strategies*, Neural Networks, vol. 7, no. 1, pp. 175-181, 1994.

[6] P. Géczy and S. Usui, *Dynamic sample selection: Theory*, IEICE Trans., Fundamentals, vol. E81-A, no. 9 September 1998.

[7] K. Hara and K. Nakayama, *Data Selection Method for Generalization of Multilayer Neural Networks*, IEICE Trans., Fundamentals,. vol. E81-A, no.3, pp.371-381 March 1998.

[8] K. Hara and K. Nakayama, *A training data selection in on-line training for multilayer neural networks*, Proc. IJCNN'98, pp. 2247-2252, Anchorage USA, May 1998.

[9] Fisher R. A., *The use of multiple measurements in taxonomic problems,* Annual Eugenics, vol. 7, Part II, pp.179-188, 1936.

[10] D. E. Rumelhart, J. L. McCelelland et. al., *Parallel Distributed Processing*, The MIT Press, vol. 1, pp. 320-330, 1986.

0-7695-0619-4/00/$10.00 (C) 2000 IEEE

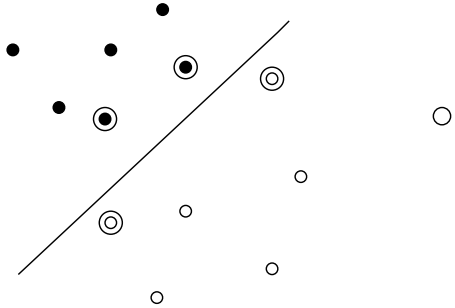

Figure 1: Example of boundary data.

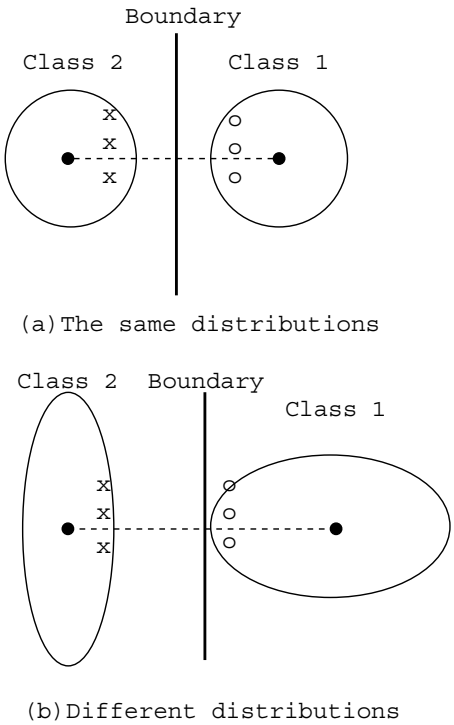

Figure 2: Boundaries of different data distribution for the same boundary data.

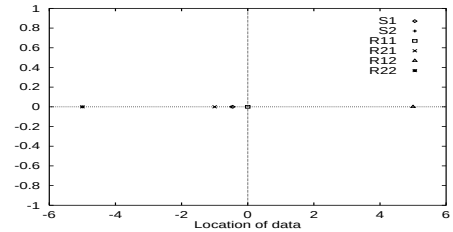

Figure 3: Random data distribution

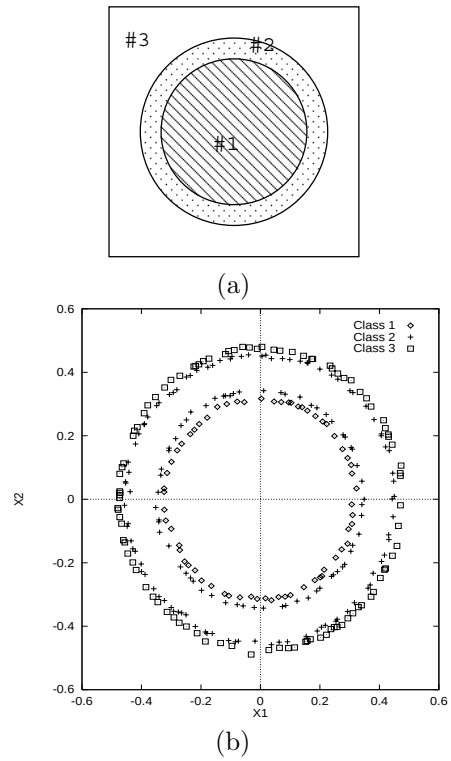

Figure 4: (a)Concept of class data distribution. (b) Selected boundary data by proposed method

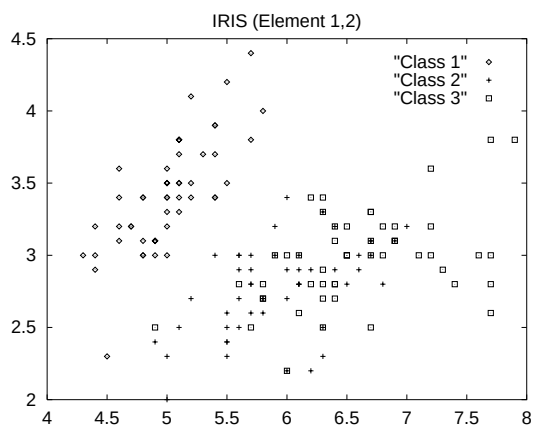

Figure 5: Data distribution of first and second element of iris data-base