Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02), Vol. 2
Lipo Wang, Jagath C. Rajapakse, Kunihiko Fukushima, Soo-Young Lee, and Xin Yao (Editors)

TRAINING RBF NEURAL NETWORKS ON UNBALANCED DATA

Xiuju Fu, Lipo Wang**, Kok Seng Chua*, Feng Chu***

*Institute of High Performance Computing
Singapore Science Park II
Singapore 117528
Email:fuxj@ihpc.a-star.edu.sg

** School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798
Email: elpwang@ntu.edu.sg
http://www.ntu.edu.sg/home/elpwang

ABSTRACT

This paper presents a new algorithm for the construction and training of an RBF neural network with unbalanced data. In applications, minority classes with much fewer samples are often present in data sets. The learning process of a neural network usually is biased towards classes with majority populations. Our study focused on improving the classification accuracy of minority classes while maintaining the overall classification performance.

1. INTRODUCTION

In neural network training, if some classes have much fewer samples compared with the other classes, the neural network system may respond wrongly for the minority classes because the overwhelming samples in the majority classes dominate the adjustment procedure in training.

Berardi and Zhang [3] proposed a cost-sensitive neural networks, by introducing different costs associated with making errors in different classes. When they calculated the sum of squared errors for the multiplayer perceptron (MLP) neural network, they multiplied each term by a class-dependent factor (cost). This idea has a much earlier origin in machine learning, that is, the loss matrix [4] which deals with different risks (costs) associated with making errors in different classes. For example, in classification of medical images, these class-dependent risk factors will need to be selected from practical experiences. By assigning larger costs to minority classes, Berardi and Zhang [3] were able to improve the classification accuracies for minority classes. In this work, as in earlier discussions on risk matrices, cost factors are selected in an ad hoc manner. In this work, we propose a method that determines these cost factors automatically such that all classes, i.e., minority and majority classes, are roughly equally important. We demonstrate this method in the case of the RBF classifier.

For unbalanced data sets, two methods had been presented in [11]. In the first method, the samples of minority classes were duplicated to increase their effects on training neural networks. In the second method, the so-called snowball method proposed in [18] for multi-font character recognition was used to improve the accuracy of the minority class [11]. In the snowball method, neural networks were first trained with the samples in the minority class, which favors the minority populations. Next, samples of majority classes were added gradually while training the neural network dynamically. The two methods were used in the MLP neural network, ART (Adaptive Resonance Theory) neural network and RBF neural network. However, it was found that the two methods mentioned above had no effect on the MLP and RBF neural networks.

As an important neural network, the RBF (radial basis function) neural network has been applied widely in various applications, such as function approximation, noisy interpolation, density estimation, optimal classification tasks and data mining tasks [2][10] [14][15][16]. It is desirable to explore a training method of RBF neural networks handling unbalanced data. The popularity of the RBF neural network is due to its fast training and its global approximation capability with local responses. Like other neural networks, constructing an RBF neural network with a compact architecture but robust for unbalanced data is a challenging task.

Here we propose an algorithm by increasing the contribution of minority samples on the MSE (mean squared error) function. The MSE function is a powerful objective

function in neural network training. When learning a data set with a minority class, the weights of a neural network will be dominated by the majority classes. For example, when using the back propagation method to train weights, the weights are updated according to the variation in the error function of the neural network [11]. It is clear that the weights obtained finally will reflect the nature of the majority classes but not much of the minority class. Thus, it motivates us to increase the magnitudes of weighted parameters of minority classes to balance the influence of the unbalanced classes, i.e., the errors brought by different classes are weighted by parameters inversely proportional to the number of samples in the classes.

This paper is organized as follows. The standard RBF neural network training algorithm is briefly reviewed in Section 2. In Section 3, we introduce our algorithm for training RBF neural networks with unbalanced data sets. A modification allowing large overlaps between clusters with the same class label when training RBF networks is presented in Section 4. In Section 5, experimental results are shown. Finally, we conclude the paper in Section 6.

2. THE STANDARD RBF NEURAL NETWORK TRAINING ALGORITHM FOR UNBALANCED DATA SETS

For a neural network classifier, its training algorithm is based on its classification performance. The MSE function is used usually as the objective function in neural networks:

$$E = \frac{1}{2}|d - Y|^2 \quad , \tag{1}$$

where d is the target vector and Y is the output vector.

In an RBF neural network, its MSE function is as follows:

$$E_0(W) = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M}\{y_m^n - t_m^n\}^2 \quad . \tag{2}$$

Here W is the weight vector. N is the number of patterns, M is the number of outputs, and K is the number of hidden units. w_{mj} is the weight connecting the jth hidden unit with the mth output unit. ϕ_j^n represents the output of the jth kernel function for the nth input pattern. t_m^n represents the target output of the mth output unit when inputting the nth pattern.

Assume there are M classes in a data set, and M output neurons in the network. The m-th output of an RBF neural network corresponding to the n-th input vector is as follows:

$$y_m^n(\vec{X^n}) = \sum_{j=1}^{K} w_{mj}\phi_j(\vec{X^n}) + w_{m0}b_m \quad . \tag{3}$$

Here $\vec{X^n}$ is the n-th input pattern vector, $m = 1, 2, ..., M$, K is the number of hidden units. w_{mj} is the weight connecting the j-th hidden unit to the m-th output node. b_m is the bias. w_{m0} is the weight connecting the bias and the m-th output node. $\phi_j^n(\vec{X^n})$ is the activation function of the j-th hidden unit corresponding to the n-th input vector.

$$\phi_j^n(\vec{X^n}) = \mathrm{e}^{\frac{||\vec{X^n} - \mathrm{C_j}||^2}{2\sigma_j{}^2}} \quad , \tag{4}$$

where $\mathrm{C_j}$ and σ_j are the center and the width for the j-th hidden unit respectively, which are adjusted during learning. When calculating the distance between input patterns and centers of hidden units, Euclidean distance measure is employed in RBF neural networks.

Substitute eq. 3 into eq. 2:

$$E_0(W) = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M}\{(\sum_{j=0}^{K} w_{mj}\phi_j^n + w_{m0}b_m) - t_m^n\}^2 \quad . \tag{5}$$

Differentiate E_0 with respect to w_{mj} and let

$$\frac{\partial E_0(W)}{\partial w_{mj}} = 0 \tag{6}$$

We obtain [4]:

$$\sum_{n=1}^{N}\{\sum_{j'=0}^{K} w_{mj'}\phi_{j'}^n - t_m^n\}\phi_j^n = 0 \quad . \tag{7}$$

Eq. 7 can be written in a matrix notation which leads to the *pseudo-inverse* for solving weights [4]:

$$(\phi^T\phi)W^T = \phi^T T \quad . \tag{8}$$

Here ϕ has dimensions $N * K$ and elements ϕ_j^n, W has dimensions $M*K$ and elements w_{mj}, and T has dimensions $N * M$ and elements t_m^n.

3. TRAINING RBF NEURAL NETWORKS ON UNBALANCED DATA SETS

In the above equations, it is observed that there is no particular attention paid on the condition that the classes in a data set are unbalanced, which may lead to unbalanced architecture in training. In our work, we add larger weights on minority classes in order to attract more attention in training for the minority members.

Assume that the number of samples in class i is N_i. The total number of samples in the data set is $N = N_1 + ... + N_i + ... + N_M$. The error function shown in eq. 5 can be written as:

$$E_0(W) = \frac{1}{2}\sum_{i=1}^{M}\sum_{n_i=1}^{N_i}\sum_{m=1}^{M}\{(\sum_{j=0}^{K} w_{mj}\phi_j^{n_i} + w_{m0}b_i) - t_m^{n_i}\}^2 \tag{9}$$

In supervised training algorithms, neural networks are construct by minimizing a neural network error function whose variables are the network weights connecting layers. For a data set with unbalanced classes, a general error function as eq. 2 or eq. 9 can not lead to a balanced classification performance on all classes in the data set because majority classes contribute more compared minority classes and therefore result in more weight adjustments. Thus, the training procedure has a bias towards frequently occurred classes.

In order to increase the contribution of minority classes in weight adjustments, we change eq. 9 to:

$$E(W) = \frac{1}{2}\sum_{i=1}^{M}\beta_i \sum_{n_i=1}^{N_i}\sum_{m=1}^{M}\{(\sum_{j=0}^{K} w_{mj}\phi_j^{n_i} + w_{m0}b_i) - t_m^{n_i}\}^2 \tag{10}$$

where

$$\beta_i = \frac{N}{N_i}, \quad i = 1, 2, ..., M \quad . \tag{11}$$

Differentiate E with respect to w_{mj}, and let

$$\frac{\partial E(W)}{\partial w_{mj}} = 0 \quad . \tag{12}$$

Substituting eq. 10 into eq. 12, we obtain:

$$\sum_{i=1}^{M}\beta_i \sum_{n_i=1}^{N_i}\{\sum_{j'=0}^{K} w_{mj'}\phi_{j'}^{n_i} - t_m^{n_i}\}\phi_j^{n_i} = 0 \quad . \tag{13}$$

We introduce a new parameter r_n replacing β_i:

$$r_n = \beta_i \quad when \quad \vec{X^n} \in A_i \quad . \tag{14}$$

A_i is class i. Substitute eq. 14 into eq. 13:

$$\sum_{n=1}^{N} r_n\{\sum_{j'=0}^{K} w_{mj'}\phi_{j'}^{n} - t_m^{n}\}\phi_j^{n} = 0 \quad . \tag{15}$$

By replacing r_n with $\sqrt{r_n}\cdot\sqrt{r_n}$, we obtain:

$$\sum_{n=1}^{N}\{\sum_{j'=0}^{K} w_{mj'}\phi_{j'}^{n}\cdot\sqrt{r_n} - t_m^{n}\sqrt{r_n}\}\phi_j^{n}\sqrt{r_n} = 0 \quad . \tag{16}$$

Similarly as stated in [4], there is the following new pseudo-inverse equation for calculating weight W:

$$(\phi^T\phi)W^T = \phi^T T \quad . \tag{17}$$

Different with the pseudo-inverse equation shown in eq. 8, here $\phi \rightarrow \phi_j^n\cdot\sqrt{r_n}$, and $T \rightarrow t_i^n\cdot\sqrt{r_n}$.

As indicated in the above equations, we have taken the unbalanced data into consideration when training RBF neural networks. Parameter r_ns introduce biased weights which is opposite to the proportions of classes in a data set. The effect of weight parameter r_n is shown in Section 5. Compared with the training method without considering the unbalanced condition in data, the classification accuracy of the minority classes are improved. We also allow large overlaps between clusters of the same class to reduce the number of hidden units [6][7].

4. LARGE OVERLAPS ALLOWED BETWEEN CLUSTERS WITH THE SAME CLASS LABEL

In applications of RBF neural networks [12][6][5][?], it is desirable to discover hidden information from data sets, and represent the discovered information in an explicit and understandable way. Thus, a compact RBF neural network classifier with high performance is preferred.

In classification, there exists two kinds of overlaps, i.e., the overlaps between different classes and the overlaps between clusters of the same class. The overlaps between different classes have been considered in RBF training algorithms. For example, overlapped receptive fields of different clusters can improve the performance of the RBF classifier when dealing with noisy data [13]. In [9] and [17], overlapping Gaussian kernel functions are created to map out the territory of each cluster with a smaller number of Gaussian functions.

In those previous methods, large overlaps between clusters of the same class are not taken into consideration. Their clusters are formed as follows. A pattern is randomly selected from the data set V as the initial center of a cluster. The radius of this cluster is chosen in such a way that the ratio between the number of patterns of a certain class (in-class patterns) and the total number of patterns in the cluster is not less than a pre-defined value θ. Once this cluster is formed, all patterns inside this cluster are "removed" from the data set and do not participate in the formation of other clusters. The value of θ is determined empirically and is related to an acceptable classification error rate. Since θ determines the radii of the clusters, it also indirectly determines the degree of overlaps between different clusters. Generally, a large θ leads to small radii of clusters, thus it leads to small overlaps between the Gaussians for different clusters and a small classification error rate for the training data set. Since a small classification error is desired, there usually exist small overlaps between the Gaussians representing the clusters.

Here is a simple example. A specified θ means that the ratio of in-class patterns in each cluster must be equal or larger than θ. In Fig. 1, suppose cluster A has been formed and its members "removed" from the data set V. Suppose pattern 2 is subsequently selected as the initial center of a new cluster and cluster B is thus formed. Clusters C through G are then formed similarly in sequence. We see that clusters B, C, and D are quite small and therefore the effective-

ness of the above clustering algorithm needs to be improved.

For allowing large overlaps between clusters of the same class, a copy V_c of the original data set V is generated first. When a qualified cluster (the ratio of in-class patterns should be higher than θ), e.g., cluster A in Fig. 2 (same as in Fig. 1), is generated, the members in this cluster are "removed" from the copy data set V_c, but the patterns in the original data set V remain unchanged. Subsequently, the initial center of the next cluster is selected from the *copy data set* V_c, but the candidate members of this cluster are patterns in the *original data set* V, and thus include the patterns in the cluster A. Subsequently when pattern 2 is selected as an initial cluster center, a much larger cluster B, which combines clusters B, C, and D in Fig. 1, can still meet the θ-criterion and can therefore be created. By allowing for large overlaps between clusters for *the same class*, we can further reduce the number of clusters substantially. This will lead to more efficient construction of RBF networks, and will be demonstrated by computer simulations in the next section.

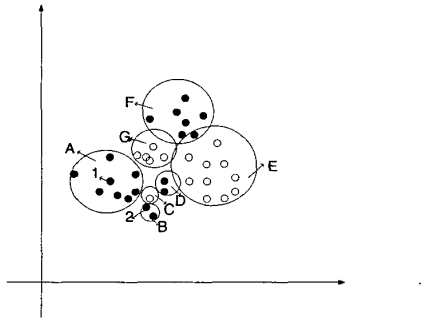

Figure 1: existing algorithms: small overlaps between clusters.

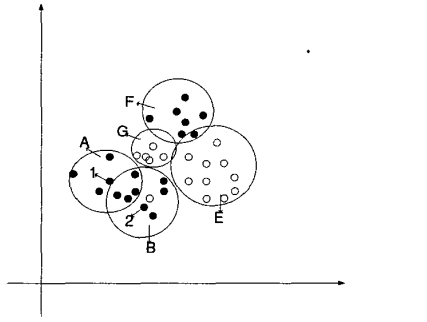

Figure 2: the modified algorithm with reduced number of clusters: small overlaps between clusters of different classes, but large overlaps between clusters of the same class.

5. EXPERIMENTAL RESULTS

Car data set [1] are used here to test our algorithm. Each data set is divided into 3 parts, i.e., training, validation, and test sets. Each experiment is repeated 5 times with different initial conditions and the average results are recorded.

Car data set is a unbalanced data set. There are 9 attributes and 2 classes (Group A and Group B) in Car data set. 4000 patterns are in training data set and 2000 patterns for testing data set. There are 507 patterns of class 1 (Group A) in training data set, and 205 patterns of class 1 in testing data set. The testing data set is divided into two subsets: validation set and testing set with 1000 patterns, respectively. Group A is the minority class. Group B is the majority class.

The classification error rates and the number of hidden units are shown comparing between small overlaps and large overlaps among clusters of the same class permitted. When allowing large overlaps among clusters of the same class, the number of hidden units is reduced from 328 to 303, and the classification error rate on the test data set is increased a little bit from 4.1% to 4.5%.

Table 1 shows the comparison of overall classification error rates between with and without considering unbalanced condition. Here large overlaps are allowed between clusters with the same class label. It is also shown in Table 1, when considering the unbalanced condition in data set, that the classification error rate of the minority class (Group A) decreases from 34.65% to 8.73%. At the same time, the error rate of the majority class increases a little bit from 1.37% to 4.1%. Since, at most cases, the minority class is embedded with important information, improving the individual accuracy of the minority class is critical.

6. SUMMARY

In this paper, we have proposed a modification to the training algorithm for the construction and training of the RBF network on unbalanced data by increasing bias towards the minority classes. Weights inversely proportional to the number of patterns of classes are given to each class in the MSE function. Experimental results show that the proposed method are effective in improving the classification accuracy of minority classes while maintaining the overall classification performance.

7. REFERENCES

[1] R. Agrawal, T. Imielinski, A. Swami, "Database mining: a performance perspective", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 6, pp. 914-925, 1993.

[2] A. Baraldi, P. Blonda, G. Satalino, A. D'Addabbo, and C. Tarantino, "RBF two-stage learning networks

Table 1: Comparison on classification error rates of the RBF neural network for each class of Car data set between with and without considering unbalanced condition when allowing large overlaps between clusters with the same class label (average results of 5 independent runs).

without considering unbalanced condition		
overall error rates		
Training set	validation set	testing set
1.89%	5.0%	4.8%
Group A		
Training set	validation set	testing set
11.69%	27.69%	34.65%
Group B		
Training set	validation set	testing set
0.77%	2.41%	1.37%
considering unbalanced condition		
overall error rates		
Training set	validation set	testing set
1.2%	5.1%	4.5%
Group A		
Training set	validation set	testing set
4.27%	4.58%	8.73%
Group B		
Training set	validation set	testing set
0.85%	5.15%	4.1%

exploiting supervised data in the selection of hidden unit parameters: an application to SAR data classification", *IEEE 2000 International Geoscience and Remote Sensing Symposium*, Vol. 2, pp. 672-674, 2000.

[3] V. L. Berardi, G. P. Zhang, "The Effect of Misclassification Costs on Neural Network Classifiers", *Decision Sciences*, Vol. 30, No. 3, USA, 1999.

[4] C. M. Bishop, *Neural network for pattern recognition*, Oxford University Press Inc., New York, 1995.

[5] X. J. Fu, L. P. Wang, "Linguistic rule extraction from a simplified RBF neural network", *Computational Statistics (a special issue on Data Mining and Statistics)*, vol. 16, no.3, pp.361-372, 2001.

[6] X. J. Fu, L. P. Wang, "Rule extraction by genetic algorithms based on a simplified RBF neural network", *Proceedings of the 2001 Congress on Evolutionary Computation*, Vol. 2, pp. 753 -758, 2001.

[7] X. J. Fu, L. P. Wang, "Data Dimensionality Reduction with Application to Simplifying RBF Network Structure and Improving Classification Performance", appear soon in *IEEE Transactions On Systems, Man, and Cybernetics-Part B: Cybernetics.*

[8] X. J. Fu, L. P. Wang, "Rule extraction using a novel gradient-based method and data dimensionality reduction", *IJCNN*, vol.2 pp. 1275 -1280, 2002.

[9] T. Kaylani and S. Dasgupta, "A new method for initializing radial basis function classifiers systems", *IEEE International Conference on Man, and Cybernetics*, Vol. 3, pp. 2584 -2587, 1994.

[10] J. Lee, C. D. Beach, N. Tepedelenlioglu, "Channel equalization using radial basis function network", *1996 IEEE International Conference on Acoustics, Speech, and Signal* Vol. 3, pp. 1719 -1722, 1996.

[11] Y. Lu, H. Guo, L. Feldkamp, "Robust neural learning from unbalanced data samples", *IEEE World Congress on Computational Intelligence*, pp. 1816 -1821, Vol. 3, 1998.

[12] K. J. McGarry, J. Tait, S. Wermter, J. MacIntyre, "Rule-extraction from radial basis function networks", *Ninth International Conference on Artificial Neural Networks(Conf. Publ. No. 470)*, Vol. 2, pp. 613 -618, 1999.

[13] P. Maffezzoni and P. Gubian, "Approximate radial basis function neural networks(RBFNN) to learn smooth relations from noisy data", *Proceedings of the 37th Midwest Symposium on Circuits and Systems*, 1994, Vol. 1, pp. 553-556.

[14] J. Moody, and C. Darken, "Fast Learning in Networks of Locally Tuned Processing Units," *Neural Computation*, 1989, Vol. 1, pp. 281-294.

[15] I. T. Nabney, "Efficient training of RBF networks for classification", *Ninth International Conference on Artificial Neural Networks (Conf. Publ. No. 470)*, Vol. 1, pp. 210 -215, 1999.

[16] W. Poechmueller, S. K. Hagamuge, M. Glesner, P. Schweikert, and A. Pfeffermann, "RBF and CBF neural network learning procedures", *1994 IEEE World Congress on Computational Intelligence*, Vol. 1, pp. 407 - 412, 1994.

[17] A. Roy, S. Govil, and R. Miranda, "An algorithm to generate radial basis function (RBF)-like nets for classification problems", *Neural networks*, Vol. 8, NO. 2, pp. 179-201, 1995.

[18] J. Wang, J. Jean, "Resolve Multifont character confusion with neural network", *Pattern Recognition*, Vol. 26, No. 1, pp. 173-187, 1993.