Joint Conference of 1996 International Computer Symposium
December 19~21, Kaohsiung, Taiwan, R.O.C.

Pattern Classification using Robust RBF Networks

Sheng-Tun Li
Department of Information Management Technology
National Institute of Technology at Kaohsiung
1 University Road
Yenchao, Kaohsiung
Taiwan, R.O.C.
Tel: +886-7-6011000 ext. 4111
Fax: +886-7-6011042
Email: stli@ccms.nitk.edu.tw

Abstract

Radial basis function networks (RBFNs) have recently attracted interest, because of their advantages over multilayer perceptrons as they are universal approximators but achieve faster convergence since only one layer of weights is required. The least squares method is the most popularly used in estimating the synaptic weights which provides optimal results if the underlying error distribution is Gaussian. However, the generalization performance of the networks deteriorates for realistic noise whose distribution is either unknown or non-Gaussian; in particular, it becomes very bad if outliers are present. In this paper we propose a positive-breakdown learning algorithm for RBFNs by applying the breakdown point approach in robust regression such that any assumptions about or estimation of the error distribution are avoidable. The expense of losing efficiency in the presence of Gaussian noise and the problem of local minima for most robust estimators has also been taken into account. The resulting network is shown to be highly robust and stable against a high fraction of outliers as well as small perturbations.

KEY WORDS: Radial basis function networks; Robust learning; Breakdown point; Least trimmed squares; Robust regression.

1 Introduction

Radial basis function networks (RBFNs), introduced by Broomhead and Lowe [1], also known as networks with locally-tuned overlapping receptive fields [2], have increasingly attracted interest for engineering applications due to their advantages over traditional multilayer perceptrons, namely simplicity and faster convergence. More importantly, RBFNs having one hidden layer are capable of universal approximation [3] as well as "almost" best approximation [4]. Given an N_D-observation data set $D = \{(\mathbf{x}_i, y_i) \mid i = 1, \cdots, N_D\}$, an N_I-N_H-1 RBFN can be regarded as a function approximator which estimates an unknown functional mapping $\lambda : \Re^{N_I} \rightarrow \Re$ such that $y_i = \lambda(\mathbf{x}_i) + \epsilon_i, \quad i = 1, \cdots, N_D$, where λ is the regression function and the error term ϵ_i is a zero-mean random variable of disturbance. The hidden layer performs a nonlinear mapping ϕ from the input space $\mathcal{X}$ to an N_H-dimensional "hidden" space Φ spanned by the transformed vector set $\{\phi(\mathbf{x}_i) \mid i = 1, \cdots, N_D\}$; *i.e.*, $\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \cdots, \phi_{N_H}(\mathbf{x})]^T$, where each nonlinear basis function $\phi_i(\mathbf{x})$ is defined by some radial basis function such as the Gaussian function. The output layer performs a biased linear combination of the radial basis functions ϕ_i to generate the function approximation $\hat{\lambda}$:

$$\hat{\lambda}(\mathbf{x}, D) = w_0 + \sum_{i=1}^{N_H} w_i \phi_i(\mathbf{x}). \qquad (1)$$

In general, training RBFNs involves two phases [2]: clustering on the hidden layer and determinating synaptic weights on the output layer in order to minimize the discrepancy between the desired response y and the predicted response $\hat{\lambda}$ for all inputs $\mathbf{x}$ with respect to some performance criterion (cost function), $\mathcal{E}$:

$$\mathcal{E}(\mathbf{w}) = \sum_{i=1}^{N_D} \rho(r_i) = \sum_{i=1}^{N_D} \rho(y_i - \hat{\lambda}(\mathbf{x}_i, \mathbf{w})), \qquad (2)$$

where ρ is the loss function which is a real-valued measure function of the residual r. If the loss function is defined by the popular squared-error L_2 criterion,

$$\mathcal{E}_{ase}(\mathbf{w}) = \frac{1}{N_D}\sum_{i=1}^{N_D}(y_i - \hat{\lambda}(\mathbf{x}_i, \mathbf{w}))^2 \qquad (3)$$

a least squares (LS) training procedure based on the average-squared error (ASE) criterion is obtained.

Since the output layer of RBFNs simply implements a multiple linear regression[1], the problem of optimizing Equation (3) can be solved by applying the singular value decomposition (SVD) procedure or by iterative gradient descent methods like least mean squares (LMS).

Training RBFNs based on the LS method provides optimal results under the assumption that the error term ϵ is Gaussian distributed[2]. However, this assumption usually fails to hold in real-world applications since either *a priori* information about the error distribution is generally unavailable or the data are contaminated by non-Gaussian noise whereby some data points fall far outside of the majority of the data so that outliers are encountered. This leads to the problems of reliability and stability of an LS estimator since training RBFNs is interpolative in nature.

Outliers may be introduced in different ways. For example, in computer vision, the outliers may be the result of clutter, large measurement errors, or impulse noise corrupting the data [5]. In general, there are two kinds of outliers, namely, *leverage points* and *vertical outliers*. Leverage points (often called *horizontal outliers*) result from contamination in the input space $\mathcal{X}$ due to some of the inputs $\mathbf{x}$ failing to obey the environmental probability rule $p(\mathbf{x})$. Contamination in the output space $\mathcal{Y}$ leads to *vertical outliers* due to the output y failing to obey the conditional probability rule $p(y|\mathbf{x})$. Both anomalies in the training set D may result in an aberrant and biased estimator since RBFN is trained to fit these significant fluctuations by interpolation instead of approximating the underlying model in an attempt to compensate for the outliers with least squared residuals. That is, RBFN is greatly sensitive to the presence of outliers.

Without doubt, outliers corresponding to large residuals should be filtered out during the training process. The problem is, given an N_D-element observation set D and a network $\mathcal{M}$ with N_W weights, how can one decide what percentage of outlying observations should be omitted? Choosing this percentage too low can make the estimator tune to the gross errors in D (overfitting), whereas choosing the percentage too high may cause some good observations to be left out (loss of efficiency). Both situations will diminish the RBFNs' generalization performance and training efficiency. Therefore, it is of great importance to justify a trade-off between robustness and efficiency and determine the the upper bound of gross errors that RBFNs can handle.

In the past decades, the theory of robust regression has provided a sound basis for dealing with deviations from the general assumption on the distribution of errors; see for example [6]. In contrast to most work on robust learning in the literature which applies the approach of an *influence function* that gives a local accurate assessment of a single outlying observation, we adopt the breakdown point approach that gives a global measure of stability in terms of the fraction of outlying data it can tolerate. We chose the approach of breakdown points mainly because of its simplicity; no *a priori* information about the error is required. It allows one to design estimators with a given breakdown point, to quantitatively compare estimators with each other, and to know what is the fraction of outliers that the estimator can handle under all conditions [7]. One notes that the breakdown point ξ^* of RBFNs is one when horizontal outliers are encountered since RBFs are bounded in general; however, they do diminish the estimation accuracy because significant residuals are produced. On the other hand, RBFNs are extreme sensitive to vertical outliers since one such outlier is sufficient to break the networks down; $\xi^* = \frac{1}{N_D}$ [8].

2 Positive-Breakdown RBFNs

In order to keep RBFNs from breaking down because of vertical outliers and improve the estimation accuracy in the presence of horizontal outliers, the least trimmed squares (LTS) method, known as an estimator with the highest possible breakdown point $\xi^* \approx 50\%$ which was proposed by Rousseeuw [9], is applied in estimating the weights in the output layer; this results in a positive-breakdown RBFN or robust RBFN (R^2BFN). Instead of minimizing the average of all squared residuals, R^2BFNs only consider the average of the smallest ordered squared residuals up to the rank q by minizing the cost function of the average-

[1] It will be a multivariate multiple linear regression if multi-output nodes are required.

[2] The RBFN is referred to as the LS-based network, henceforth.

trimmed-squared error (ATSE):

$$\mathcal{E}_{atse}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{q} r_{(i)}^2, \quad (4)$$

where $r_{(1)}^2 \leq \cdots \leq r_{(q)}^2 \leq \cdots \leq r_{(N_D)}^2$ are ordered squared residuals and q is the number of residuals that needs to be taken into account at the true weight vector $\mathbf{w}$. It follows from theoretical analyses of RBFNs [8] and of robust regression [9, 6] that R^2BFNs reach their optimal breakdown point

$$\xi^*_{R^2BFN}(\hat{\lambda}, D) = \frac{\lfloor \frac{N_D - N_W}{2} \rfloor + 1}{N_D} \quad (5)$$

when $q = \lfloor \frac{N_D}{2} \rfloor + \lfloor \frac{N_W+1}{2} \rfloor$, where $N_W = N_H + 1$ is the number of weights including the one between the biased unit and the output unit.

The cost function in Equation (4), as for most high-breakdown estimators, is non-convex and can have several local minima [6]. To improve the performance in terms of learning speed and probability of convergence, R^2BFNs are enhanced by applying a normalized steepest descent method [10] with the following weight update rule:

$$\triangle \mathbf{w} = -\eta \frac{\partial \mathcal{E}_{atse}}{\partial \mathbf{w}} \Big/ \left\| \frac{\partial \mathcal{E}_{atse}}{\partial \mathbf{w}} \right\|^2, \quad (6)$$

where η is the learning rate. In order to accommodate both requirements of robustness against outliers and of efficiency in the presence of Gaussian noise, R^2BFNs are further improved by adaptively adjusting q according to

$$q(t+1) = \lfloor \tau(t) N_D \rfloor + \lfloor (1 - \tau(t))(N_W + 1) \rfloor, \quad (7)$$

where $\tau \in [\frac{1}{2}, 1]$ determines the proportion of observations involved in error backpropagation and is a function of the commonly used criterion *normalized root-mean-squared error* (NRMSE) on an uncontaminated test set V since it keeps track of the generalization ability of the network:

$$\tau(e) = 0.5 \exp(-\frac{\nu}{e^2}) + 0.5, \quad (8)$$

where ν is a small positive real number.

These two modifications of R^2BFNs together result in a new network, the so-called adaptive R^2BFN (AR^2BFN for short).

3 A Robust Dichotomizer

In order to validate the effectiveness of the proposed network, we discuss some experimental results. We first consider dichotomies on two-dimensional feature spaces defined by the robust dichotomizer, our AR^2BFN network, in the presence of vertical and horizontal outliers.

3.1 Vertical Outliers

In this experiment, a two-dimensional feature (or input) space over the rectangle $[-1, 1] \times [-1, 1]$ is assumed. The decision boundary is defined by the discriminant function

$$\mathcal{L}(x_1, x_2) = x_2 - \frac{(sin(\pi x_1) + cos(\pi x_1))}{2} \quad (9)$$

which separates the feature space $\mathcal{X}$ into two non-overlapping regions

$$\mathcal{R}_1 \equiv \{(x_1, x_2) \mid \mathcal{L}(x_1, x_2) \geq 0\}$$

and

$$\mathcal{R}_2 \equiv \{(x_1, x_2) \mid \mathcal{L}(x_1, x_2) < 0\}.$$

An uncontaminated training set $D = \{(\mathbf{x}_i, y_i) \mid i = 1, \cdots, 120\}$ is constructed by uniformly sampling 60 data points from each region, with $y_i = 0.5$ and $y_i = -0.5$, defining the classes $\mathcal{C}_1$ and $\mathcal{C}_2$. A 220-element validation set V is similarly formed. Figure 1 plots the scatter diagram for the input patterns in D.

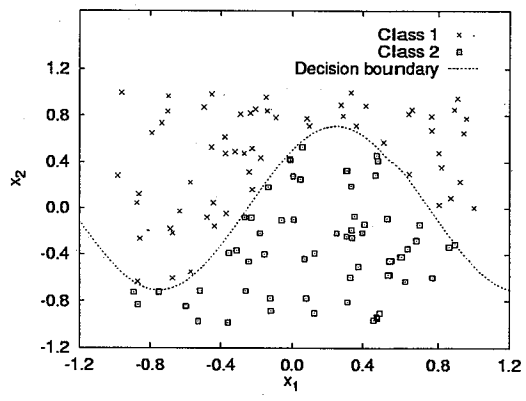

Figure 1: The scatter diagram for feature vectors in the uncontaminated training set D with 60 points in each region separated by the discriminant function $\mathcal{L}$.

Since the output component y of each training datum differing from x is discrete, a suitable vertical

outlier model is randomly mislabelling training data as in [11, 12]. For this, a corrupted training set D' is constructed by deliberately flipping the labels for six and seven sample points in region $\mathcal{R}_1$ and $\mathcal{R}_2$ as shown in Figure 2.

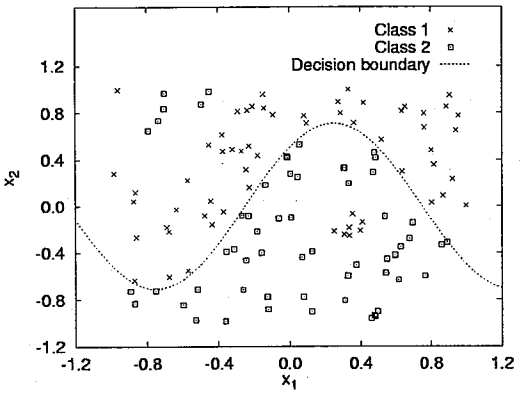

Figure 2: The scatter diagram for the contaminated training set D' in which six and seven points in regions $\mathcal{R}_1$ and $\mathcal{R}_2$ are deliberately mislabelled.

The conventional RBFN and the robust dichotomizer AR^2BFN are trained by the set D' and validated by the set V with network topology 2-10-1, learning rate 0.005 and $\nu = 0.06$ for AR^2BFN. Therefore, the maximal number of outliers that can be tolerated by this AR^2BFN is 54 since $\xi^* = 0.458$ according to Equation (5). Figure 3 depicts the evolution of the classification rate on the validation set in terms of learning epochs in which our AR^2BFN achieves 97% in 4250 epochs in comparison to 90% in 9000 epochs for the RBFN. Figures 4 and 5 illustrate the generalization performance for RBFN and AR^2BFN.

One can see that the influence of vertical outliers has been successfully diminished by AR^2BFN, compared to RBFN. It is worth noting that the outliers appearing in region $\mathcal{R}_2$ are scattered more densely than other neighboring points (see Figure 2) and hence have stronger impact; this causes a few of the points in Region $\mathcal{R}_1$ close to the boundary to be misclassified. Therefore, the impact of moderate outliers depends on their scattering density and location.

3.2 Horizontal Outliers

For horizontal outliers, we use two lightly overlapping classes $\mathcal{C}_1$ and $\mathcal{C}_2$. Both classes are represented by two

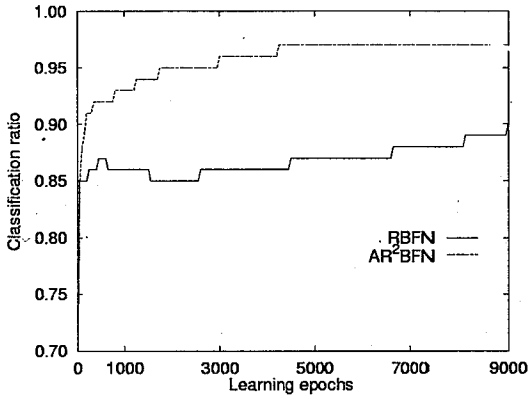

Figure 3: The evolution of classification rate in terms of learning epochs for AR^2BFN and RBFN in the presence of vertical outliers for the pattern classification problem.

bivariate normal distributions with the identity covariance matrix, $N([0,0]^T, \mathbf{I})$ and $N([2,2]^T, \mathbf{I})$. An uncontaminated set D and validation set V are created by uniformally sampling 60 and 110 points respectively from each class as shown in Figures 6 and 7.

To introduce realistic outliers, the class $\mathcal{C}_1$ is assumed to be contaminated by outliers with Tukey's *contaminated normal distribution* [13], which has a mixture density

$$(1-\zeta)N(\mathbf{0},\mathbf{I}) + \zeta N(\mathbf{0}, 3^2\mathbf{I}). \qquad (10)$$

In other word, each point in $\mathcal{C}_1$ is contaminated with probability ζ. Figure 8 plots the scatter diagram of the contaminated training set D' composed of class $\mathcal{C}_1$ with $\zeta = 33\%$ and $\mathcal{C}_2$. Notice that 20 points in $\mathcal{C}_1$ have been spread out toward the region of class $\mathcal{C}_2$.

For comparison, RBFN and AR^2BFN are trained with 2-5-1 network structures, learning rate $\eta = 0.0001$ and $\nu = 0.06$. Both networks rapidly achieve good learning performance with a classification rate of 91% for RBFN and 93% for AR^2BFN within 350 and 160 epochs, respectively. Figures 9 and 10 show the scatter diagrams of the classification results for both networks. One notes that the difference between both networks in generalization performance is quite small in the area close to the classification boundary since protection is provided by bounded Gaussian hidden units as we pointed earlier. On the other hand, the misclassification made by RBFN on the comparably far removed points confirms our conjecture (*e.g.*, the pattern at $(4.89, 0.19)$); that is, the impact of outliers

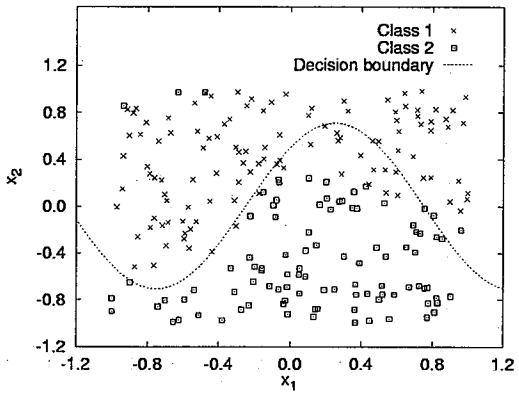

Figure 4: The classification result obtained by RBFN, with a 90% classification rate.

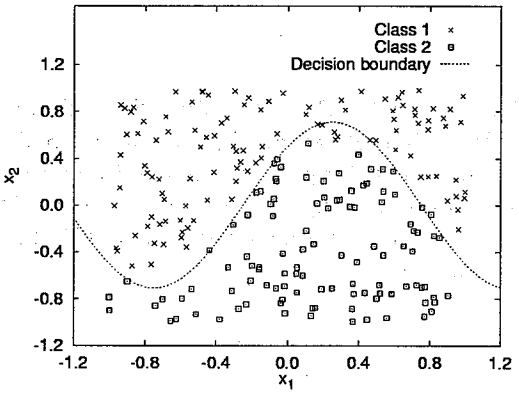

Figure 5: The classification result obtained by AR^2BFN, with a 97% classification rate.

is determined by the location and density of outliers. A more comprehensive study on the effect of sample density affecting the performance of learning by locally tuned networks can be found in [14], in which the factors of unit noise and receptive field size, shape, and overlap are also considered.

4 Conclusions

The work presented in this paper was motivated by the need for developing a robust radial basis function network (RBFN) that can handle more realistic noise in which outliers or gross errors may occur. Unlike most studies which use the influence function approach, we applied the breakdown point approach in

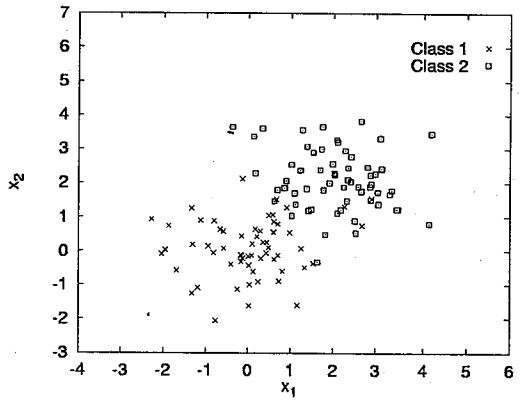

Figure 6: The scatter diagram for a 120-element uncontaminated training set D for classes C_1 and C_2 uniformly sampled from two equiprobable bivariate normal distributions $N([0,0]^T, \mathbf{I})$ and $N([2,2]^T, \mathbf{I})$.

developing robust RBFNs such that neither *a prior* information about the error distribution nor estimating it is required. In particular, an adaptive robust learning algorithm based on the least trimmed squares method was proposed in order to improve the robustness against gross errors, probability of escaping local minima, and training efficiency in the presence of Gaussian noise. The effectiveness of the resulting positive-breakdown RBFN has been validated by performing experiments on pattern classification.

References

[1] D. S. Broomhead and D. Lowe, "Multivariable Functional Interpolation and Adaptive Networks", *Complex Systems,* Vol. 2, pp. 321-355, 1988.

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

[3] J. Park and I. W. Sandberg, "Universal Approximation Using Radial-Basis Function Networks", *Neural Computation,* Vol. 3, pp. 246-257, 1991.

[4] T. Poggio and F. Girosi, "Networks and the Best Approximation Property", *Biological Cybernetics,* Vol. 63, pp. 169-176, 1990.

[5] P. Meer, D. Mintz, A. Rosenfeld, and D. Y. Kim, "Robust Regression Methods for Computer Vi-

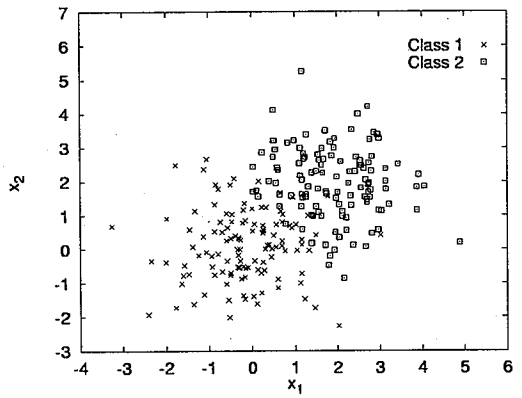

Figure 7: The scatter diagram for an uncontaminated validation set V with 120 points for each class having bivariate normal distribution.

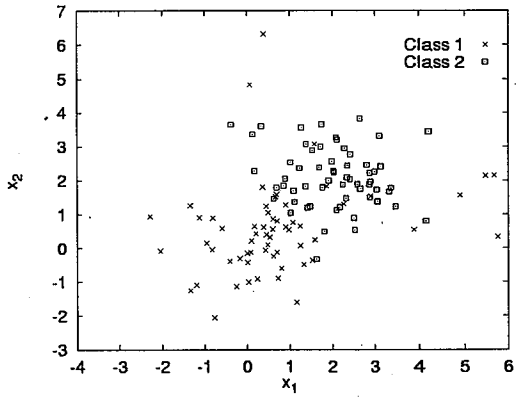

Figure 8: The scatter diagram for the contaminated training set D'. $\mathcal{C}_1$ is corrupted by the contaminated normal distribution with probability $\zeta = 33\%$; it results in 20 points in $\mathcal{C}_1$ being spread out toward the region of $\mathcal{C}_2$.

sion: A Review", *Internation Journal of Computer Vision,* Vol. 6, pp. 59-70, 1991.

[6] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection,* John Wiley & Sons, New York, 1987.

[7] L. Mili, *Personal communication,* 1995.

[8] S.-T. Li, "Spatiality and Stochasticity in Artificial Neural Networks", Ph.D. Dissertation, University of Houston, May 1995.

[9] P. J. Rousseeuw, "Least Median of Squares Regression", *Journal of the American Statistical Association,* Vol. 79, pp. 871-880, 1984.

[10] A. G. Parlos, B. Fernandez, A. F. Atiya, J. Muthusami, and W. K. Tsai, "An Accelerated Learning Algorithm for Multilayer Perceptron Networks", *IEEE Trans. on Neural Networks,* Vol. 5, pp. 493-497, 1994.

[11] S. Geman and E. Bienenstock, "Neural Networks and the Bias/Variance Dilemma", *Neural Computation,* Vol. 4, pp. 1-58, 1992.

[12] P. Burrascano, "Robust Learning in the Presence of Outliers", *1994 International Symposium on Artificial Neural Networks,* Taiwan, pp. 31-35, 1994.

[13] J. B. McDonald and S. B. White, "A Comparison of Some Robust, Adaptive, and Partially Adaptive Estimators of Regression Models", *Econometric Reviews,* Vol. 12, No. 1, pp. 103-124, 1993.

[14] B. W. Mel, "How Receptive Field Parameters Affect Neural Learning", in J. Moody, S. J. Hanson, and R. Lippmann (eds.), *Advances in Neural Information Processing Systems 3,* Morgan Kaufmann Publishers, San Mateo, CA, pp. 757-763, 1991.

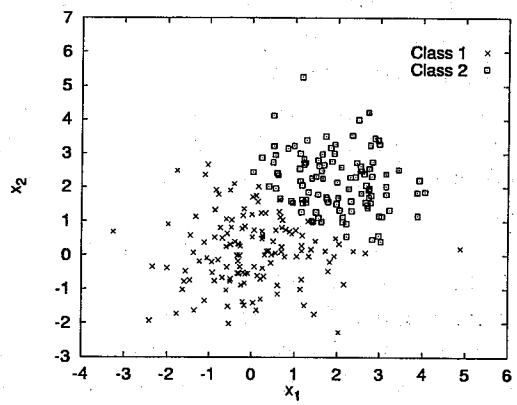

Figure 9: The classification result obtained by RBFN on the test data shown in Figure 7, with a 91% classification rate. Notice that the pattern at (4.89, 0.19) has been misclassified.

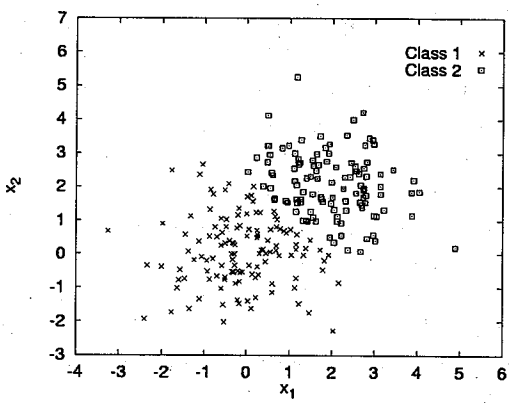

Figure 10: The classification result obtained by AR^2BFN on the test data shown in Figure 7, with a 93% classification rate. Note that the pattern at (4.89, 0.19) has been correctly classified.