Proc. Intentional Joint Conference on Neural Networks, Vol., May 5-9, 1998, Anchorage, Alaska.

Rival Penalized Competitive Learning, Finite Mixture, and Multisets Clustering

Lei Xu
Computer Science and Engineering Department
The Chinese University of Hong Kong, Shatin, NT, Hong Kong, P.R.China

Abstract— **Rival Penalized Competitive Learning (RPCL) can automatically select the number of clusters during learning via penalizing the rival in competition. The original adaptive RPCL algorithm is proposed for clusters of spherical shapes and its performance will degenerate considerably when the clusters are of complicated shapes. In this paper, adaptive RPCL learning has been extended to solve this problem via the finite mixture modeling and multi-sets modeling, respectively. Moreover, two general competition types are suggested, called Type A and Type B. The Type B RPCL with both the finite mixture modeling and multi-sets modeling includes the original RPCL as a special case. The experiments have shown that both Type A and Type B RPCL work well and improved the original RPCL considerably for clusters of complicated shapes and strong overlapping. Moreover, the Type B RPCL is the best in automatic selection of correct number of clusters.**

Keywords— **Clustering Analysis, Rival Penalized, Competitive Learning, Non-spherical Shape.**

1. Introduction

Given a data set $D_x = \{x_i\}_{i=1}^N$, the task of partitioning D_x into k clusters is a classical problem in the literature of statistics and pattern recognition, usually called cluster analysis [1], [2], [3]. A well known formulation of this task is to use k vectors $\{m_j^*\}_{j=1}^k$, called centers or code-vectors to represent the k clusters such that a sample x_i is classified into the y-th cluster when $I(y|x_i) = 1$, according to:

$$I(y|x_i) = \begin{cases} 1, & \text{if } y = arg\ min_j \|x_i - m_j\|^2 \\ 0, & \text{otherwise} \end{cases}, \quad (1)$$

with $\{m_j^*\}_{j=1}^k$ obtained by

$$Min_{\{m_j\}_{j=1}^k} E_{MSE},$$
$$E_{MSE} = \frac{1}{N}\sum_{j=1}^{k}\sum_{i=1}^{N} I(y|x_i)\|x_i - m_j\|^2. \quad (2)$$

This formulation is called the *Mean Square Error (MSE)* clustering analysis or vector quantization. It is typically implemented by the well known k-means algorithm or the LBG algorithm, which is a two-step iterative procedure, starting from an initial guess on either $\{m_j\}_{j=1}^k$ or $\{I(y|x_i)\}_{i=1}^N$:

$$Step\ \ 1\ :\ update\ \{I(y|x_i)\}_{i=1}^N\ by\ eq.(1),\ get$$
$$\alpha_j = \frac{1}{N}\sum_{i=1}^{N} I(y|x_i),$$
$$Step\ \ 2\ :\ update\ \ m_j = \frac{1}{\alpha_j N}\sum_{i=1}^{N} I(y|x_i)x_i.$$

Equivalently, many of the so called competitive learning algorithms in the literature of neural networks can be regarded as adaptive variants of the above algorithm for MSE clustering (e.g., see the Ref. List in [11]).

The algorithms in this formulation all have two serious limitations. The first one is that the number k of clusters must be pre-known and fixed. E_{MSE} monotonically decreases with increasing k, and thus cannot detect a correct k. However, a bad estimate of k can cause serious problems as stated in [11]. To the best of our knowledge, the selection of a correct k remains an important open problem. It is usually tackled by some heuristic techniques, e.g., ISODATA [1], [2], [3], and Rival Penalized Competitive Learning (RPCL) [11]. The other limitation is that the formulation implies that samples come from a mixture of k Gaussian densities with equal proportion and equal variance $\sigma^2 I$. This special case deviates from many practical situations. In the literature, the so called Mahalanobis distance clustering or elliptic clustering attempts to overcome this limitation.

In this paper, the original adaptive RPCL learning has been extended to clustering of data clusters with complicated shapes, via the finite mixture modeling [6], [7] and multi-sets modeling [10], respectively. Moreover, two general competition types are suggestion, called Type A and Type B. The experiments will be made to demonstrate the success of the proposed extensions.

This project was supported by the HK RGC Earmarked Grants CUHK250/94E and CUHK484/95E and by Ho Sin-Hang Education Endowment Fund for Project HSH 95/02.

2. Adaptive RPCL Learning with Finite Mixture Modeling

Rival Penalized Competitive Learning (RPCL) is a heuristic competitive learning algorithm proposed for clustering with a favorable feature that it can drive away extra neurons during learning with automatic number selection[11]. The key idea of RPCL is that for each input x_i not only the winner is learned to approach x, but also the second winner (rival) is de-learned away from x_i for a bit. That is,

$$\begin{aligned} &m_c^{(t+1)} = m_c^{(t)} + \gamma_c(x_i - m_c), \\ &\quad for\ c = arg\ min_j\ f_j\|x_i - m_j\|^2 \\ &m_r^{(t+1)} = m_r^{(t)} - \gamma_r(x_i - m_r), \\ &\quad for\ r = arg\ min_{j\neq c}\ f_j\|x_i - m_j\|^2 \\ &m_j^{(t+1)} = m_j^{(t)}, \quad for\ j \neq c,\ j \neq r, \end{aligned} \tag{3}$$

with $\gamma_c > \gamma_r$ and f_j being the frequency that m_j wins the competition up to now. Many experiments have shown that RPCL works well although there is no theoretical justification. In Xu(1995a), via an incremental EM algorithm, we have justified the RPCL's heuristic use of de-learning together with learning[9].

Two types of general forms of RPCL are proposed in [8] based on the following finite mixture:

$$P(x) = \sum_{j=1}^{k} \alpha_j P_j(x, \theta_j), \tag{4}$$

That is, we consider two types of competitions:
Competition Type A (MAP-RPCL):

$$\begin{aligned} c =&\ arg\ max_j \alpha_j P_j(x, \theta_j), \\ r =&\ arg\ max_{j\neq c} \alpha_j P_j(x, \theta_j). \end{aligned} \tag{5}$$

Competition Type B:

$$\begin{aligned} c =&\ arg\ max_j \alpha_j \ln P_j(x, \theta_j), \\ r =&\ arg\ max_{j\neq c} \alpha_j \ln P_j(x, \theta_j). \end{aligned} \tag{6}$$

Then, for each input x we update parameters by

$$\begin{aligned} \theta_c^{(t+1)} &= \theta_c^{(t)} + \gamma_c \frac{\partial \log P(x, \theta_c)}{\partial \theta_c^{(t)}} \\ \theta_r^{(t+1)} &= \theta_j^{(t)} - \gamma_r \frac{\partial \log P(x, \theta_r)}{\partial \theta_r^{(t)}}, \\ \theta_j^{(t+1)} &= \theta_j^{(t)}, \quad y \neq c,\ y \neq r. \end{aligned} \tag{7}$$

The idea of this RPCL is to further enforce the winner component in the mixture while weaken its rival component in responsible to the current input x. As a result, input data set will be divided into groups according the above two types of decision.

Type A is equivalent to $c = arg\ max_j\ P(j|x)$ because $P(j|x) = \alpha_j P_j(x, \theta_j)/P(x)$, which is usually called Baysian or Maximum Posterior (MAP) Decision, so we call the corresponding RPCL by MAP-RPCL. Also, $\alpha_c P_c(x, \theta_c)$ is the largest component in the mixture density $P(x)$, which represents the average likelihood.

For Type B, $\alpha_c \ln P_c(x, \theta_c)$ is the largest component in the mixture log-likelihood function, which represents the average log-likelihood. The two types become the same when α_j is equal for each j. They are different in accounting the effect of α_j, which can be simply approximated by f_j–the frequency that m_j wins the competition up to now. For a smaller α_j, the neuron's competitive ability becomes weaker in Type A, but strong in Type B. In the other word, Type B is more conscience.

When $P_j(x, \theta_j)$ is a gaussian $G(x, m_j, \Sigma_j)$, we have:
Competition Type A (MAP-RPCL):

$$\begin{aligned} c =&\ arg\ min_j d_j, \\ r =&\ arg\ max_{j\neq c} d_j, \\ d_j =&\ (x - m_j)\Sigma_j^{-1}(x - m_j) + \log|\Sigma_j| - 2\log \alpha_j \end{aligned} \tag{8}$$

Competition Type B:

$$\begin{aligned} c =&\ arg\ min_j d_j, \\ r =&\ arg\ max_{j\neq c} d_j, \\ d_j =&\ \alpha_j[(x - m_j)\Sigma_j^{-1}(x - m_j) + \log|\Sigma_j|]. \end{aligned} \tag{9}$$

and from eq.(7) the updating on parameters m_j becomes exactly eq.(3). We can let the updating for α_j be simply approximated by f_j–the frequency that m_j wins the competition up to now. In the special case that $\Sigma_j = I$ of spherical shape. This Type B returns to exactly the original RPCL. In the general case for Σ_j, we need to update it too, for which a batch way algorithm is suggested in [8].

In this paper, we propose the following adaptive updating for

$$\begin{aligned} &\Sigma_c^{(t+1)} = (1 - \gamma_c)\Sigma_c^{(t)} + \gamma_c(x_i - m_c)(x_i - m_c)^T, \\ &\Sigma_y^{(t+1)} = \Sigma_y^{(t)}, for\ \ y \neq c. \end{aligned} \tag{10}$$

3. Adaptive RPCL Learning with Multi-Sets Modeling

Given $\mathcal{X} = \{x_i\}_{i=1}^N$, we restate the multisets modeling learning [10] for the problem of parametric clustering in a general paradigm that consists of two subtasks. One is partitioning $\mathcal{X}$ into K sets (or called clusters) $C_1, \cdots, C_j$. The second is to model each set C_j by a parametric expression, denoted by $C_j(\theta_j)$ or simply θ_j. Suppose that the error or distortion of using this model to represent a sample x_i is denoted by

$\varepsilon(x_i, \theta_j) \geq 0$, our purpose is to assign each sample x_i into a C_j according to an indicator $I(j|x_i) = 1$ if C_j wins the following competition

$$I(y|x_i) = \begin{cases} 1, & \text{if } y = arg\ min_j\ \varepsilon(x_i, \theta_j) \\ 0, & \text{otherwise} \end{cases}, \quad (11)$$

such that the parameters $\{\theta_j\}_{j=1}^K$ are obtained by

$$Min_{\{\theta_j\}_{j=1}^K} \sum_{j=1}^K E(j, \theta_j),$$
$$E(j, \theta_j) = \frac{1}{N} \sum_{i=1}^N I(k|x_i)\varepsilon(x_i, \theta_j). \quad (12)$$

In general, again we propose to solve this problem by an extension of RPCL algorithm. That is, get the winner by $k = arg\ min_j\ f_j\varepsilon(x_i, \theta_j)$ with f_j being the frequency that C_j wins in the past, and get the rival by $r = arg\ min_{j \neq k}\ f_j\varepsilon(x_i, \theta_j)$; Then, update :

$$\theta_c^{(t+1)} = \theta_c^{(t)} - \gamma_c \frac{\partial \varepsilon(c, \theta_c)}{\partial \theta_c^{(t)}}$$
$$\theta_r^{(t+1)} = \theta_r^{(t)} + \gamma_r \frac{\partial \varepsilon(r, \theta_r)}{\partial \theta_r^{(t)}},$$
$$\theta_j^{(t+1)} = \theta_j^{(t)}, \quad y \neq c,\ y \neq r. \quad (13)$$

This Multi-Sets Modeling paradigm actually unifies various types of clustering via using different parametric models for C_j and different measures for $\varepsilon(x_i, \theta_j)$:

Type 1. Conventional Mean Square Error(MSE) Clustering (having linear boundaries between clusters): Each C_j is of spherical shape in equal scale and the task is to locate C_j's center point. Thus, we have $\theta_j = m_j$ and the error measure is $\varepsilon(x_i, \theta_j) = \|x_i - m_j\|^2$. This kind of clustering has been widely studied already. In fact, the above algorithms in this case reduce to the case of eq.(1) and eq.(2) by the K-means algorithm eq.(3) or the original RPCL algorithm eq.(3).

Type 2. Weighted MSE (or called Mahalanobis distance) Clustering (having quadratic boundaries between clusters): C_j is of elliptic shape, with its location given by a point m_j and its *shape + orientation* by a positive definite matrix Σ_j. That is, $\theta_j = \{m_j, \Sigma_j\}$. The error $\varepsilon(x_i, \theta_j) = (x_i - m_j)^T\Sigma_j^{-1}(x_i - m_j)$ is the Mahalanobis distance between x_i and m_j. When $\Sigma_j = \sigma_j^2 I$, every cluster is still of spherical shape but now can be of different scale factor σ_j. When Σ_j is a general positive diagonal matrix, every cluster is a canonical ellipse. Furthermore, when Σ_j is general positive definite symmetric matrix, every cluster is a general ellipse in any orientation and scale.

If Σ_j is known in advance, its implementation is the same as *Type 1* except of using $\varepsilon(x_i, \theta_j) = (x_i - m_j)^T\Sigma_j^{-1}(x_i - m_j)$ to replace $\varepsilon(x_i, \theta_j) = \|x_i - m_j\|^2$ in competition. When Σ_j is unknown, we can use eq.(10) for its updating.

Type 3. Local Least Mean Square Error Reconstruction (LMSER) or Local PCA Clustering (having quadratic boundaries between clusters): C_j is a line passing through a point m_j and along the direction $w_j - m_j$ which is the principal axis of C_j with maximum variance. Thus, $\theta_j = \{m_j, w_j\}$. The error $\varepsilon(x_i, \theta_j) = \|(x_i - m_j) - (w_j - m_j)(w_j - m_j)^T(x_i - m_j)\|^2$ is the Square Reconstruction Error $\|x_i - \hat{x}_i\|^2$ by the orthogonal project $\hat{x}_i = m_j + (w_j - m_j)(w_j - m_j)^T(x_i - m_j)$ of $x_i - m_j$ on the principal direction $w_j - m_j$. An equivalently measure can also be $\varepsilon(x_i, \theta_j) = \|x_i - m_j\|^2 - \frac{[(w_j - m_j)^T(x_i - m_j)]^2}{\|w_j - m_j\|^2}$.

Since the solution of this clustering is that m_j is the mean of C_j and $w_j - m_j$ is the principal axis of C_j with $\Sigma_j(w_j - m_j) = \lambda_{max}(w_j - m_j)$. One adaptive way for solving it is to cooperate in Oja rule [4] and its anti-hebbian variant[12]:

$$\begin{aligned}
&m_c^{(t+1)} = m_c^{(t)} + \gamma(x_i - m_c^{(t)}), \\
&y_c = (x_i - m_c^{(t+1)})^T(w_c^{(t)} - m_c^{(t+1)}), \\
&w_c^{(t+1)} = w_c^{(t)} + \\
&\gamma_c y_c[(x_i - m_c^{(t+1)}) - y_c(w_c^{(t)} - m_c^{(t+1)})], \\
&m_r^{(t+1)} = m_r^{(t)} - \gamma(x_i - m_r^{(t)}), \\
&w_j^{(t+1)} = w_j^{(t)}, j \neq c, \\
&m_j^{(+1)} = m_j^{(t)}, \quad j \neq c, \quad j \neq r.
\end{aligned} \quad (14)$$

Type 4. Local MCA Clustering (having quadratic boundaries between clusters): C_j is a hyperplane passing through a point m_j with its normal direction $w_j - m_j$ which is the minor axis of C_j with the minimum variance. Thus, $\theta_j = \{m_j, w_j\}$. The error measure is $\varepsilon(x_i, \theta_j) = \frac{[(w_j - m_j)^T(x_i - m_j)]^2}{\|w_j - m_j\|^2}$.

Its adaptive implementation is quite similar to *Type 3*. More specifically, the part for updating m_j is still the same. However, updating for w_j are replaced by

$$\begin{aligned}
&y_c = \frac{(x_i - m_c^{(t+1)})^T(w_c^{(t)} - m_c^{(t+1)})}{\|w_c^{(t)} - m_c^{(t+1)}\|}, \\
&w_c^{(t+1)} = w_c^{(t)} - \\
&\gamma y_c[(x_i - m_c^{(t+1)}) - y_c\frac{(w_c^{(t)} - m_c^{(t+1)})}{\|w_c^{(t)} - m_c^{(t+1)}\|}], \\
&y_r = \frac{(x_i - m_r^{(t+1)})^T(w_r^{(t)} - m_r^{(t+1)})}{\|w_r^{(t)} - m_r^{(t+1)}\|},
\end{aligned} \quad (15)$$

$$\begin{aligned}
&w_r^{(t+1)} = w_r^{(t)} + \\
&\gamma y_r[(x_i - m_r^{(t+1)}) - y_r \frac{(w_r^{(t)} - m_r^{(t+1)})}{\|w_r^{(t)} - m_r^{(t+1)}\|}], \\
&m_j^{(+1)} = m_j^{(t)}, w_j^{(t+1)} = w_j^{(t)}, \quad j \neq c, \ \ j \neq r.
\end{aligned}$$

Type 5. Local Principal Subspace Clustering (having quadratic boundaries between clusters): C_j is a r-dimensional subspace spanned by $W_j = [w_j^{(1)}, \cdots w_j^{(r)}]$ with the subspace's origin located at a point m_j. Thus, $\theta_j = \{m_j, W_j\}$. The error measure is $\varepsilon(x_i, \theta_j) = \|(I - P)(x_i - m_j)\|^2$ with $P = W_j(W_j^t W_j)^{-1} W_j^t$ for a nonorthogonal matrix W_j or equivalently $P = W_j W_j^T$, $W_j^T W_j = I$ for an orthogonal matrix W_j.

Its adaptive implementation is an extension of *Type 3* to higher dimension cases. Similarly, we use Oja subspace rule [5] and its anti-Hebbian variant for its updating:

$$\begin{aligned}
&y_c = (W_c(t) - m_c^{(t+1)}[1, \cdots, 1])^T (x_i - m_c^{(t+1)}), \\
&W_c^{(t+1)} = W_c^{(t)} + \\
&\gamma_c[(x_i - m_c^{(t+1)}) - (W_c^{(t)} - m_c^{(t+1)}[1, \cdots, 1])y_c]y_c^T, \\
&y_r = (W_r^{(t)} - m_r^{(t+1)}[1, \cdots, 1])^T (x_i - m_r^{(t+1)}), \\
&W_r^{(t+1)} = W_r^{(t)} - \qquad\qquad\qquad\qquad (16) \\
&\gamma^r[(x_i - m_r^{(t+1)}) - (W_r^{(t)} - m_r^{(t+1)}[1, \cdots, 1])y_r]y_r^T, \\
&m_j^{(+1)} = m_j^{(t)}, W_j^{(t+1)} = W_j^{(t)}, \quad j \neq c, \ \ j \neq r.
\end{aligned}$$

Type 6. Nonlinear Manifold Clustering: C_j is represented by some nonlinear manifold and $\varepsilon(x_i, \theta_j)$ is some kind of projection error to this manifold.

Type 7. Hybrid Clustering: $C_1, \cdots, C_j$ and its related error are not necessary to be represented in the same type, but can be a mixture of more than one of the above types.

Through the following Gibbs distribution

$$P_j(x, \theta_j) = \frac{1}{Z_j} exp(-\lambda_j \varepsilon(j, \theta_j)), \lambda_j > 0 \qquad (17)$$

we can link the Multi-Sets Modeling to the finite mixture eq.(4). Moreover, in the special case that Z_j, λ_j are the same for all j or if we can approximately ignore the difference between Z_j, λ_j for different j. Then, the above discussed RPCL learning algorithms for each type of multi-sets are the same as the TYPE B RPCL learning in Sec.2.

4. Experiments

In the experiments, as shown in Fig.1, 2-dimensional data sets with different shapes and degree of overlapping are used. They all have four clusters and each contains 500 data points. The clusters in data set 1 and 2 are very clearly separated with the later one contains ellipse-shaped clusters. For data set 3, there are two ellipse shaped clusters overlapping together on the left and those in data set 4 also similar but with overlapping clusters on each side. In this paper, only the results on comparison with (i) the original RPCL eq.(3) (denoted by *Point case* since each cluster is simply presented by its center point), (ii) the TYPE A RPCL learning eq.(3) (simply denoted by Type A), eq.(8) & eq.(10) and (iii) the TYPE B RPCL learning eq.(3), eq.(10) & eq.(10) (simply denoted by Type B), are considered.

(1) *Cluster Number Detection* In this part, the cluster number detection ability of the algorithms are studied. For all the tests, five units are initialized. All the mean vectors randomly initialized with range [-1,1], and all the covariance matrices (if any) are positive diagonal random matrix with entries ranged in [0,1]. The selected mean movements (learning paths of mean vectors) by the original RPCL and TYPE B RPCL on the data set 2 are shown in Fig. 2 and Fig.3. The successful rates on four data sets within ten tests are summarized in Table 1. It can be seen that although all the algorithms work well for Data set 1 and Data set 2, as the overlap increases, the original RPCL degenerates quickly and usually will fail. Moreover, Type B RPCL performs the best and can always success.

(2) *The Confusion Matrices of the successful clustering.* In this part, assume that we already know the number of clusters and fix the number of neurons on the correct number $k = 4$. The confusion matrices on clustering results on the four data sets by the three algorithms are shown in Tables 3, 4, 5 & 6 respectively. The averaged percentage of correctness in clustering each data set are shown in Table 2. It can be observed again, that the performance of the original RPCL degenerates considerably as the overlaps between clusters increases. However, Type A RPCL and TYPE B RPCL both work very well even for the very strongly overlapped Data set 4, and there is no obvious differences between the performance of Type A and Type B. It means that both types of RPCL work well if the number of cluster is set correctly.

5. Conclusions

For clusters of complicated shapes with strong overlapping, the performance of original RPCL algorithm degenerates considerably, however both Type A and Type B RPCL work well and improved the original RPCL considerably. Moreover, the Type B RPCL is the best in automatic selection of correct number of clusters.

The author would like to thank Mr. Wing-kai Lam for the help in preparing Figs.1-3 and Tabs.1-6.

REFERENCES

[1] Duda, R.O, & Hart, P.E. (1972), *Pattern Classification and Science Analysis*, John Wiely.

[2] Jain, A.K., and R.C.Dubes (1988), *Algorithm for Clustering Data*, Prentice-Hall.

[3] P.A.Devijier and J. Kittler, *Pattern Recognition - A Statistical Approach*, Pretice- Hall, Englewood-Cliffs, 1982.

[4] E.Oja(1982), "A simplified neuron model as a principal component analyzer", *Journal of Mathematical Biology, 16*, 1982, 267-273.

[5] E.Oja(1989), "Neural networks, principal components, and subspaces", *Int. J. Neural Systems 1*, 1989, 61-68.

[6] R.A., Redner, and H.F. Walker," Mixture densities, maximum likelihood, and the EM algorithm", *SIAM Review 26*, 1984, 195-239.

[7] L. Xu, "Bayesian Ying-Yang Machine, Clustering and Number of Clusters", *pattern Recognition Letters*, in press, 1997.

[8] L. Xu, "Bayesian-Kullback YING-YANG Learning Scheme: Reviews and New Results", *Proc. Intl Conf. on Neural Information Processing (ICONIP96)*, Sept. 24-27, Springer-Verlag, pp59-67(1996).

[9] L. Xu,"YING-YANG Machine: a Bayesian-Kullback scheme for unified learnings and new results on vector quantization", Keynote talk, Proc. Intl Conf. on Neural Information Processing (ICONIP95), 1995, pp977-988.

[10] L. Xu," A Unified Learning Framework: Multisets Modeling Learning", Invited paper, Proc. World Congress On Neural Networks, July 17-21, 1995, Washington, DC, Vol.I, pp35-42, 1995. A early version is partly in Proc. 1994 IEEE Intl Conf. on Neural Networks, June 26-July 2, Orlando, Florida, Vol.I, pp315-320, 1994.

[11] Xu, L., Krzyzak, A. and Oja, E.(1993), " Rival Penalized Competitive Learning for Clustering Analysis, RBF net and Curve Detection", *IEEE Trans. on Neural Networks*, Vol.4, No.4, pp636-649, 1993.

[12] Xu, L, Oja, E. & Suen, C. Y., "Modified Hebbian learning for curve and surface fitting"., *Neural Networks 5*, 441-457(1992).

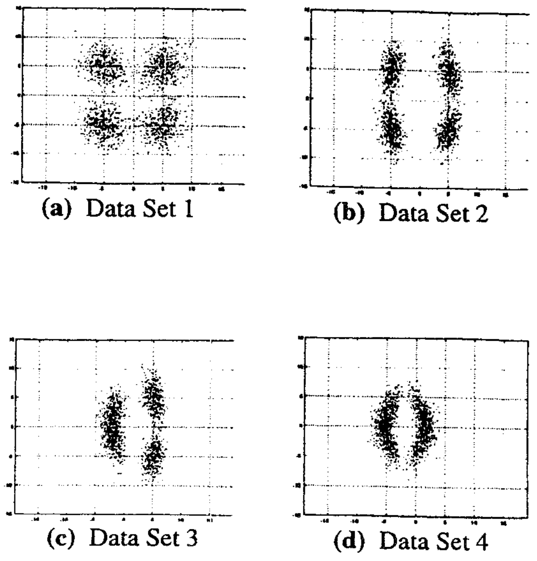


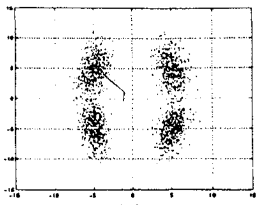
Fig. 1 The four testing data set

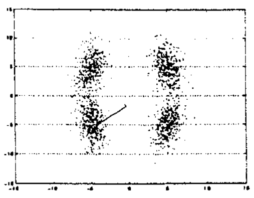
	Data set 1	Data set 2	Data set 3	Data set 4
Point case	100%	100%	20%	10%
Type A	100%	100%	60%	60%
Type B	100%	100%	100%	100 %

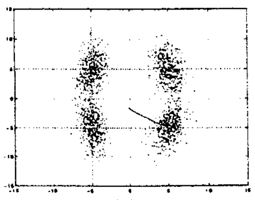
Table 1 The percentage correctness by the RPCL algorithms in detecting the cluster numbers inside the four data set in ten tests

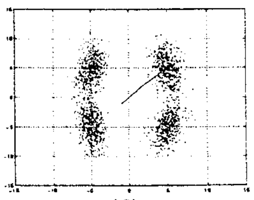
	Data set 1	Data set 2	Data set 3	Data set 4
Point case	99.75%	98.05%	88.20%	81.05%
Type A	98.35%	98.00%	94.20%	93.15%
Type B	98.95%	98.55%	94.20%	93.35%

Table 2 The average percentage of classification error by various algorithms on the 4 data set


(a)


(b)


(c)


(d)

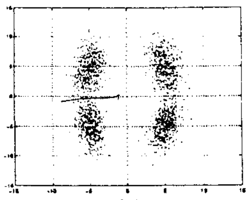
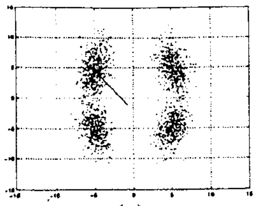
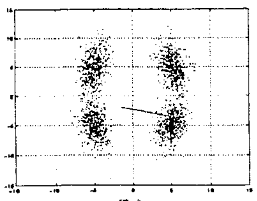
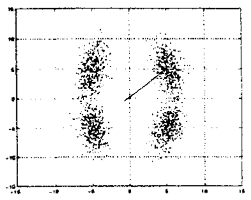
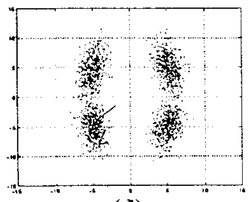

(e)

Fig. 2 The Five neuron movements in learning by the original RPCL on Data Set 2


(a)


(b)


(c)


(d)

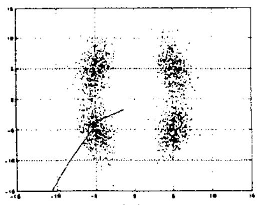

(e)

Fig. 3 The five neuron movements in learning by Type-B RPCL on Data Set 2

	θ_1	θ_2	θ_3	θ_4
C1	99.6	0	0.4	0
C2	0	100	0	0
C3	0	0.2	99.8	0
C4	0.4	0	0	99.6

(a) Original RPCL

	θ_1	θ_2	θ_3	θ_4
C1	98.0	1.0	0	1.0
C2	0.6	98.2	1.2	0
C3	0	0.2	98.4	1.4
C4	0.8	0	0.4	98.8

(b) Type-A RPCL

	θ_1	θ_2	θ_3	θ_4
C1	98.4	0.8	0	0.8
C2	0.4	98.6	1.0	0
C3	0	0.2	99.0	0.8
C4	0.2	0	0	99.8

(c) Type-B RPCL

Table 3 The confusion matrix on the clustering results on Data Set 1

	θ_1	θ_2	θ_3	θ_4
C1	94.4	5.6	0	0
C2	0.2	99.8	0	0
C3	0	0	98.4	1.6
C4	0	0	0.4	99.6

(a) Original RPCL

	θ_1	θ_2	θ_3	θ_4
C1	98.2	0	1.8	0
C2	0.4	99.6	0	0
C3	0	0	95.0	5.0
C4	0	0	0.8	99.2

(b) Type-A RPCL

	θ_1	θ_2	θ_3	θ_4
C1	97.0	3.0	0	0
C2	0.8	99.2	0	0
C3	0	0	98.0	2.0
C4	0	0	0	100

(c) Type-B RPCL

Table 4 The confusion matrix on the clustering results on Data Set 2

	θ_1	θ_2	θ_3	θ_4
C1	77.2	20.2	5.6	0
C2	18.0	77.4	0	4.6
C3	0	0	98.6	1.4
C4	0	0	0.4	99.6

(a) Original RPCL

	θ_1	θ_2	θ_3	θ_4
C1	91.4	1.8	6.8	0
C2	11.8	87.8	0	0.4
C3	0	0	98.6	1.4
C4	0	0.2	0.8	99.0

(b) Type-A RPCL

	θ_1	θ_2	θ_3	θ_4
C1	86.2	13.2	0.6	0
C2	2.2	94.6	0	3.2
C3	0	0	99.6	0.4
C4	0	0	3.6	96.4

(c) Type-B RPCL

Table 5 The confusion matrix on the clustering results on Data Set 3

	θ_1	θ_2	θ_3	θ_4
C1	85.8	14.0	0.2	0
C2	21.0	78.8	0	0.2
C3	0.2	0	86.8	13.0
C4	0	0	23.2	76.8

(a) Original RPCL

	θ_1	θ_2	θ_3	θ_4
C1	97.6	1.4	1.0	0
C2	10.2	89.2	0.2	0.4
C3	0	0	97.8	2.2
C4	0	0	12.0	88.0

(b) Type-A RPCL

	θ_1	θ_2	θ_3	θ_4
C1	94.6	5.0	0.4	0
C2	0.4	97.4	0	2.2
C3	0	0	95.6	4.4
C4	0	0	2.2	97.8

(c) Type-B RPCL

Table 6 The confusion matrix on the clustering results on Data Set 4