Fuzzy algorithms for Robust Clustering

Tai-Ning Yang, Sheng-De Wang* and Shi-Jim Yen**
Dept. of Computer Science, Chinese Culture University,
Dept. of Electrical Engineering, National Taiwan University* and
Dept. of Computer Science and Information Engineering, National Dong Hwa University**

Abstract

In this paper, we consider the issue of clustering when outliers exist. The outlier set is defined as the complement of the data set. Following this concept, a specially designed fuzzy membership weighted objective function is proposed and the corresponding optimal membership is derived. Not like the membership of fuzzy c-means, the derived fuzzy membership does not reduce with the increase of the cluster number. A hard clustering algorithm alleviating the prototype under-utilization problem is also derived. Artificially generated data are used for comparison.

1 Introduction

Clustering algorithms try to partition a set of unlabeled input vectors into a number of subsets(clusters) such that data in the same subset are more similar to each other than to data in other subsets. There are two kinds of unsupervised clustering algorithms: hierarchical versus partitional. Hierarchical clustering generates a sequence of nested partitions from the proximity matrix. In social, biological, and behavioral sciences, hierarchical clustering techniques are popular because of the necessity to produce taxonomies. Partitional clustering is used frequently in the engineering applications such as data compression and image segmentation. A single partition of the data is generated in partitional clustering. Clustering algorithms can also be divided into two types: hard versus fuzzy. Hard (crisp or conventional) clustering assigns each input vector to exactly one cluster. In fuzzy clustering, a given pattern does not necessarily belong to only one cluster but can have varying degrees of memberships to several clusters. Many clustering algorithms could be found in the the related books [1]- [4].

We propose a new family of clustering algorithms called fuzzy robust clustering (FRC) with the consideration of outliers. The size of updating prototype is independent of the number of prototypes and the influence of outliers is reduced.

The other parts of this chapter are organized as follows. In section 2, we review fuzzy c-means (FCM). The following section introduces our algorithm called fuzzy robust clustering (FRC). The prototype under-utilization problem of LVQ is explained and the proposed hard robust clustering (HRC) algorithms are introduced in section 4. Section 5 demonstrates the simulations. Section 6 contains the conclusions.

2 Fuzzy c-means and the outliers

Let $X = \{x_1, x_2, ..., x_n\} \subset R^m$ denote the input sample set and c denote the number of nodes in the output layer. The prototypes $V = \{v_1, v_2, ..., v_c\}$ are cluster centers, where $v_i \in R^m$ for $1 \leq i \leq c$. Let U denote the membership matrix. The objective function in *Fuzzy C-Means* (FCM) [5] is:

$$E(V, X) = \sum_{i=1}^{c} \sum_{j=1}^{n} {u_{ij}}^m \|x_j - v_i\|^2. \qquad (1)$$

In the above equation, $m \in [1, \infty)$ is a weighting factor called fuzzifier. The elements u_{ij} in the matrix U satisfy the following constraints: :

constraint 1. $u_{ij} \in [0, 1]$ for all i,j.

constraint 2. $0 < \sum_{j=1}^{n} u_{ij} < n$ for all i,j.

constraint 3. $\sum_{i=1}^{c} u_{ij} = 1$ for all j.

Constraint 3 is a probabilistic premise. If it is relaxed without modifying (1), a trivial solution in which all memberships are zero will be produced.

Bezdek [5] applied the technique of Lagrange multiplier and derived the optimal membership function,

$$u_{ij} = (\sum_{r=1}^{c} \frac{\|x_j - v_i\|^{2/(m-1)}}{\|x_j - v_r\|^{2/(m-1)}})^{-1}. \qquad (2)$$

The problem of Fuzzy c-means when outliers exist could be found in the paper [6].

3 Fuzzy robust clustering algorithms

Now, we propose our objective function. Assume that there is a noise cluster outside each data cluster. The fuzzy complement of u_{ij}, $f(u_{ij})$ may be interpreted as the degree to which x_j does not belong to the i-th data cluster. Thus the fuzzy complement can be viewed as the membership of x_j in the noise cluster with the distance η_i. The general form of the proposed objective function is:

$$FE(V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n}\left\{{u_{ij}}^m d^2(x_j,v_i) + (f(u_{ij}))^m \eta_i\right\}. \tag{3}$$

$d(x_j, v_i)$ is the distance measure from the data point x_j to prototype v_i. A special version of (3) with the derived algorithm called *possibilistic c-means* (PCM) was proposed by R. Krishnapuram and J. M. Keller [6]. They used the standard fuzzy complement, $f(u_{ij}) = 1 - u_{ij}$.

To minimize (3), various methods such as the genetic approach, the simulated annealing, and the alternating optimization could be used [7]. We choose the alternating optimization for its popularity . The alternating optimization is driven by necessary conditions of minimization. We select $f(u_{ij}) = 1 + u_{ij} * log(u_{ij}) - u_{ij}$ because of the simplicity of its derivatives. It is easy to prove that this setting of $f(u_{ij})$ satisfies the axioms of fuzzy complements, boundary conditions and monotonicity [8]. For the ease of the following discussion, we set $m = 1$. Thus, the proposed fuzzy objective function becomes:

$$FE(V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij} d^2(x_j,v_i) + \tag{4}$$

$$\sum_{i=1}^{c}\sum_{j=1}^{n}(1 + u_{ij} * log(u_{ij}) - u_{ij})\eta_i. \tag{5}$$

The distance is usually set as the Euclidean distance, that is $d(x_j, v_i) = \|x_j - v_i\|$. First, we compute the gradient of FE with respect to u_{ij}. By setting $\frac{\partial FE(V,X)}{\partial u_{ij}} = 0$, we get

$$u_{ij} = exp(-\frac{\|x_j - v_i\|^2}{\eta_i}). \tag{6}$$

Substituting this membership back and after simplification, we get

$$FE(V,X) = \sum_{i=1}^{c}\eta_i - \sum_{i=1}^{c}\sum_{j=1}^{n} exp(-\frac{\|x_j - v_i\|^2}{\eta_i})\eta_i. \tag{7}$$

The above reformulation is based on the method proposed by Hathaway and Bezedek [10]. Following the multidimensional chain rule, when x_j is the current input data the gradient of FE with respect to v_i is

$$\frac{n\partial FE(V,X)}{\partial v_i} = u_{ij}\Big(\frac{\partial\|x_j - v_i\|^2}{\partial v_i}\Big) \tag{8}$$

$$= -2u_{ij}(x_j - v_i). \tag{9}$$

In (6), η_i plays the role of a normalization parameter for the distance $\|x_j - v_i\|^2$ for the membership u_{ij}.

Fuzzy Robust Clustering (FRC) algorithm:

Input: all of the training feature vector set $X = \{x_1, x_2, ..., x_n\}$ and the number of clusters c.

Output: the final prototypes of clusters $V = \{v_1, v_2, ..., v_c\}$.

Procedure:

step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$.

step 2. Set the initial prototype set $V = \{v_1, v_2, ..., v_c\}$ with a strategy.

step 3. Compute $\alpha_t = \alpha_0(1 - t/T)$ and adjust η_i with a strategy.

step 4. Sequentially take every sample x_j from X and update each prototype with $v_i^{new} = v_i^{old} + \alpha_t(x_j - v_i^{old})(exp(-\frac{\|x_j - v_i\|^2}{\eta_i}))$.

step 5. Add 1 to t and repeat step 3 through step 5, until t is equal to T.

η_i plays an important role in (6) and determines the mobility of the corresponding prototype. There are many strategies for the adjusting of η_i. We propose two strategies. One is initializing η_i with the result of another clustering algorithm. The other is initializing the prototypes at different places with larger distances between each other and setting a smaller $\alpha_0 \in (0, 1]$ and a larger T. In step 5, η_i is adjusted by the rule:

$\eta_i = min_j\Big\{\|v_j - v_i\|^2\Big\}, j \neq i$. The concept is to set η_i with the minimum influence on the other prototype. This is often referred as the alternation optimization approach [7]. It is hard to find the membership function that is the critical point of the following objection function:

$$FE(V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n} {u_{ij}}^m d^2(x_j, v_i) + \quad (10)$$

$$\sum_{i=1}^{c}\sum_{j=1}^{n}(1 + u_{ij} * log(u_{ij}) - u_{ij})\eta_i. \quad (11)$$

Alternatively, we could seek other optimization approaches. The proposed membership (6) can also be derived from the M estimator approach in the field of robust statistics [11]. The M estimator approach tries to minimize the function:

$$ME(v) = \sum_{i=1}^{n}\theta(r_i), \quad (12)$$

where $r_i = x_i - v$ is the residuals and $\theta(\dot{})$ is a symmetric positive-definite function. Various functions for $\theta(\dot{})$ can be selected to reduce the influence of large residuals. We propose $\theta(r_i) = exp(-r_i^2)$. The necessary condition for the minimum of (12) is $\frac{\partial ME(v)}{\partial v} = 0$. Thus,

$$\sum_{i=1}^{n}(x_i - v)exp(-r_i^2) = 0. \quad (13)$$

We get

$$v = \frac{\sum_{i=1}^{n} exp(-r_i^2)x_i}{\sum_{i=1}^{n} exp(-r_i^2)}. \quad (14)$$

Equation 14 shows v is a weighted mean of x. The only difference between the weighting function and the proposed fuzzy membership function is just the normalization. Thus FRC algorithm can be viewed as c M estimators work independently and simultaneously.

Note that the membership function (6) is a special case of

$$u_{ij} = exp(-(\frac{\|x_j - v_i\|^2}{\eta_i})^p), \quad (15)$$

that is used by Runkler and Bezdek without indicating it is the objective function [7]. Here, we propose the corresponding fuzzy objective function,

$$FF(V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij} d^p(x_j, v_i) + \quad (16)$$

$$\sum_{i=1}^{c}\sum_{j=1}^{n}(1 + u_{ij} * log(u_{ij}) - u_{ij})\eta_i{}^p. \quad (17)$$

Following the alternation optimization approach, we compute the gradient of FF with respect to u_{ij}. By setting $\frac{\partial FF(V,X)}{\partial u_{ij}} = 0$, we get

$$u_{ij} = exp(-(\frac{\|x_j - v_i\|^2}{\eta_i})^p). \quad (18)$$

The gradient of FF with respect to v_i is

$$\frac{n\partial FF(V,X)}{\partial v_i} = \quad (19)$$

$$-2u_{ij}pd^{(p-1)}(x_j, v_i)(x_j - v_i). \quad (20)$$

To ensure the convergence of the algorithm, the product $pd^{(p-1)}(x_j, v_i)$ is removed. Since $pd^{(p-1)}(x_j, v_i)$ is always positive, the removal does not affect the modification direction of v_i. We call the corresponding algorithm FRC2. FRC2 is the same as FRC except the computation of the membership function in step 4.

step 4. Sequentially take every sample x_j from X and update each prototype with $v_i^{new} = v_i^{old} + \alpha_t(x_j - v_i^{old})(exp(-(\frac{\|x_j - v_i\|^2}{\eta_i})^p))$.

It may be interesting to compare the proposed function(15) and the membership function proposed by Raghu Krishnapuram et al. [9]

$$u_{ij} = \frac{1}{1 + (\frac{\|x_j - v_i\|^2}{\eta_i})^p}. \quad (21)$$

The membership values of these two functions relative to the normalized distance $\frac{\|x_j - v_i\|^2}{\eta_i}$ are shown in Fig. 1. p is set from 1 to 10, respectively. Properly used, the exponential parameter p can produce various membership functions. The two functions have the similar form because their objective functions share the similar design logic.

Runkler and Bezdek [7] proposed a fuzzy clustering scheme called alternating cluster estimation with the above two membership functions as the membership function tool bars.

4 Prototype under-utilization problem of LVQ and hard robust clustering algorithms

Learning vector quantization (LVQ) is a neural network based method to find a good set of

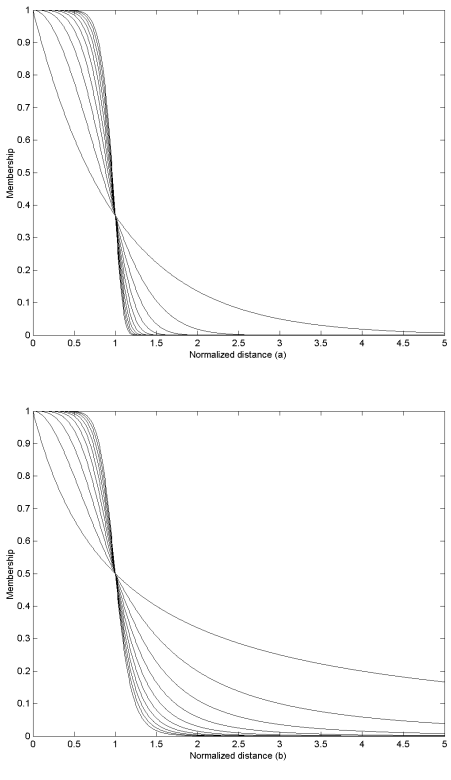

Figure 1: Plot of the membership generated with different p. The upper plot is the proposed membership function and the lower plot is proposed by Raghu Krishnapuram et al.

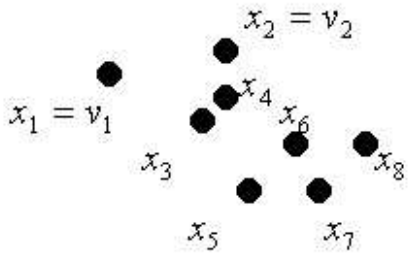

Figure 2: An initialization problem of LVQ and SHCM.

prototypes to store as the reference set for representing the cluster structure of the input data. Let $X = \{x_1, x_2, ..., x_n\} \subset R^m$ denote the input sample set and c denote the number of nodes in the output layer. The objective of LVQ is to minimize the following function :

$$E(V, X) = \sum_{i=1}^{c} \sum_{j=1}^{n} w_{ij} \|x_j - v_i\|^2, \qquad (22)$$

where $w_{ij} = 1$ if v_i is the winner else $w_{ij} = 0$.

It was found by Grossberg [12], Rumelhart, and Zipser [13] that traditional hard clustering algorithms like the sequential hard c-means (SHCM) and learning vector quantization (LVQ) suffer from a severe initialization problem. If the initial values of the prototypes are not in the convex hull formed by the input data, the clustering algorithms may not produce meaningful results. This is called prototype under-utilization or dead units problem since some prototypes, called dead units, may never win the competition. The cause of this problem is that these algorithms update only the winning prototype for every input.

We use the following example to show that LVQ suffers from the initialization problem. Suppose that the input data $X = \{x_1, x_2, ..., x_8\}$. There are two classes $X_1 = \{x_1, x_2, x_3, x_4\}$ and $X_2 = \{x_5, x_6, x_7, x_8\}$ in the data set. $v_{i,j}$ is the prototype v_i in the iteration j. The initial positions of two prototypes $v_{1,0}$ and $v_{2,0}$ are at x_1 and x_2 as shown in Fig. 2. Since $v_{2,0}$ is closer to other six input data than $v_{1,0}$, only $v_{2,0}$ is updated. $v_{1,0}$ never gets a chance to win. This result is not desirable because it is only a local optimum.

A popular approach proposed to solve the initialization problem is frequency sensitive competitive learning (FSCL) [14]. In FSCL, each prototype incorporates a count of the number of times it has been the winner. The distance measure is modified to give prototypes with a lower count value a chance to win the competition.

FSCL algorithm:

step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$ and winning count $u_i = 0$.

step 2. Set the initial prototype set $V = \{v_1, v_2, ..., v_c\}$ with some strategy.

step 3. Compute $\alpha_t = \alpha_0(1 - t/T)$.

step 4. For $k = 1, 2, .., n$, find the winning neuron v_i, such that $u_i * \|x_k - v_i\| =$

$min_{1\le j\le c}\{u_j * \|x_k - v_j\|\}$. Update the winner and winning count with $v_i^{new} = v_i^{old} + \alpha_t(x_k - v_i^{old})$ and $u_i = u_i + 1$.

step 5. Add 1 to t and repeat step 3 through step 5, until t is equal to T.

According to the analysis by Pal, Bezdek, and Tsao [15], the problem has two causes: (1) an improper initialization of prototypes and (2) only one prototype is updated in each input. To solve (1), some researchers suggest to initialize prototypes with random input vectors. This approach reduces the probability of falling into this situation but does not eliminate it.

Pal, Bezdek, and Tsao [15] suggest an algorithm called GLVQ that gives the full membership to the winner and the partial membership to the loser. We found the inconsistent situation produced by GLVQ for a certain scaling of input data and proposed a modified algorithm called generalized competitive clustering (GCC) [16]. This situation is also analyzed by Gonzalez et al. [17] Another modified algorithm called GLVQ-F is designed by Nikhil et al. [18] , where a modified function is proposed. Since GLVQ-F uses the membership derived from the FCM, it has the same disadvantages of FCM when the cluster number is large.

Following the concept of GLVQ-type algorithm, we modify the FRC algorithm to give the winner full membership. That is, $\eta_i = \infty$ if the i-th prototype is the winner. While the losers use the membership (6) with the finite η. We call the proposed algorithm *hard robust clustering* (HRC). Like the situation in FRC, the initial value and adjustment of η in HRC is very important. In the current version, we let the losers use the same η and reduce it as the training process prodceeds. The initial value of η is set as a multiple of the squared variance of the training data. Depending on the distribution of the data and a priori information, η can also be set with other strategies.

Hard Robust Clustering (HRC) algorithm:

Procedure:

step 1. Initially set the iteration count $t = 1$, iteration bound T, learning coefficient $\alpha_0 \in (0, 1]$.

step 2. Set the initial prototype set $V = \{v_1, v_2, ..., v_c\}$ with a strategy.

step 3. Compute $\alpha_t = \alpha_0(1 - t/T)$ and $\eta = K * var^2 * \eta(1 - t/T)$. K is a constant and var is the variance of the data set X.

step 4. For $j = 1, 2, .., n$, find the winning neuron v_i such that $\|x_j - v_i\| = min_{1\le k\le c}\{\|x_j - v_k\|\}$. Update the winner with $v_i^{new} = v_i^{old} + \alpha_t(x_k - v_i^{old})$. Update the others with $v_k^{new} = v_k^{old} + \alpha_t(x_j - v_i^{old})(exp(-\frac{\|x_j - v_i\|^2}{\eta_t}))$.

step 5. Add 1 to t and repeat step 3 through step 5, until t is equal to T.

5 Simulations

5.1 Comparison of LVQ, FSCL and HRC with same number of data in four clusters

Input data: There are four clusters of samples, and each cluster has 100 samples from four Gaussian distributions centered at $(0.1, 0)$, $(0, 0.1)$, $(-0.1, 0)$, $(0, -0.1)$, marked by Asterisk.

Initialization: Initial positions of four prototypes are all at (0.25, 0.25). We set $T = 40$, $c = 4$, $K = 3$ and $\alpha_0 = 0.8$.

Results:

LVQ: The final positions of four prototypes trained by LVQ are shown in Fig. 3. Two prototypes stay at initial positions.

HRC and FSCL: All prototypes are used by HRC and FSCL and the final positions are near the actual centroids as shown in Fig. 4 and Fig. 5.

Discussions: The results correspond to the designing purpose of the HRC and FSCL. Note that the performance of HRC is not sensitive to the setting of the constant K. A choice of K in [1,10] produces the similar result.

5.2 Comparison of FSCL and HRC with different number of data in four clusters

This simulation is designed to test the robustness of FSCL and HRC when the number of data in each cluster is different. As shown in Fig. 5, FSCL performs well. But if we change the number of data in clusters to 50, 50, 200, 200, the final centroids trained by FSCL shift along the direction from the sparse cluster to the dense cluster

as shown in Fig. 6. This is caused by the winning frequency used in FSCL. HRC does not use the frequency as one of the training coefficients, therefore the produced prototypes will not be affected by the number of data in clusters as shown in Fig. 7. Note that in the above simulations, the results of FRC depend on the initialization. It performs like HRC if the prototypes are initialized properly or it fails like LVQ.

5.3 Comparison of FRC and HRC

Input data: There are four clusters of samples, and each cluster has 100 samples from four Gaussian distributions marked by dot. There are 100 outliers marked by "+".

Initialization: Initial positions of four prototypes are at (0.2, 0.2), (0.2, -0.2), (-0.2, 0.2) and (-0.2, -0.2). We set $T = 40$, $c = 4$, $K = 3$ and $\alpha_0 = 0.2$. The initial value of v_i and η_i of FRC is set with the result of HRC.

Results: As shown in Fig. 8, the final positions of prototypes in HRC are greatly affected by the outliers. The final prototypes of FRC are near the actual centroids as shown in Fig. 9.

Discussions: Following the progress of the training process, η of the losers reduces and the behavior of HRC is more and more like the LVQ. Thus the final prototypes are affected by the outliers. Since all prototypes in FRC use the fuzzy membership, the influence field is limit. FRC alleviates the effect from the outliers but its prototype mobility is also restricted.

5.4 Comparison of PCM and FRC

Although the PCM and FRC2 have similar membership function, they still perform differently in some simulations.

Input data: There are two clusters of samples. The left cluster has 100 and the right cluster has 300 samples from two Gaussian distributions marked by dot.

Initialization: Initial positions of two prototypes are at (0, 0). We set $T = 40$, $p = 2$, $c = 2$ and $\alpha_0 = 0.2$.

Results: As shown in Fig. 10, the final positions of prototypes in FRC are satisfied. The final prototypes are near the actual centroids. While in the PCM, the left prototype is affected by the right cluster as shown in Fig. 11.

Discussions: From the memberships shown in Fig. 1, when the normalized distance is greater than 1, PCM membership is usually greater than FRC membership. Thus the left prototype of PCM is affected by the right cluster and the greater bias appears.

6 Conclusions

Through the definition of outliers as the fuzzy complement, we release the probabilistic constraint and propose a family of robust clustering objective functions. A detailed example of such algorithms is given.

Compared with other clustering algorithms, FRC and FRCSS alleviate the influence from the outliers. We also modify FRC to be a hard robust clustering algorithm for solving the prototype under-utilization problem. As supported by the experiments, the proposed algorithms not only perform well under the assumed model, but also produce a satisfactory result under the noisy environment.

References

[1] K. Nadler and E. P. Smith, *Pattern Recognition Engineering,* John Wiley & Sons, 1993.

[2] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition,* John Wiley & Sons, 1992.

[3] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data.* Eaglewood Cliffs, NJ: Prentice-Hall. 1988.

[4] J. C. Bezdek and S. K. Pal, Eds., *Fuzzy Models for Pattern Recognition.* New York: IEEE Press. 1992.

[5] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms.* New York: Plenum, 1981.

[6] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE Trans. Fuzzy sys.,* vol. 1, pp. 98–110, 1993.

[7] T. A. Runkler and J. C. Bezdek, "Alternating cluster estimation: a new tool for clustering and function approximation," *IEEE Trans. Fuzzy sys.,* vol. 7, no. 4, pp. 377–393, 1999.

[8] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic.* Prentice-Hall Inc., pp. 51–61, 1995.

[9] R. Krishnapuram, O. Nasraoui, and H. Frigui, "The fuzzy c spherical shells algorithm: a new approach," *IEEE Trans. Neural Net.,* vol. 3, no. 5, pp. 663–671, 1992.

[10] R.
J. Hathaway and J. C. Bezdek,"Optimization of clustering criteria by reformulation," *IEEE Trans. Fuzzy sys.,* vol. 3, no. 2, pp. 241–245, 1995.

[11] P. J. Huber, *Robust Statistics.* New York: Wiley, 1981.

[12] S. Grossberg, "Competitive learning: From interactive activations to adaptive resonance," *Cognitive Science,* vol. 11, pp. 23–63, 1987.

[13] D. Rumelhart and D. Zipser, "Feature discovery by competitive learning," *Cognitive Science,* vol. 9, pp. 75–112, 1985.

[14] D. Desieno, "Adding a conscience to competitive learning," in *Proc. IEEE Int. Conf. Neural Net.,* vol. I, 1988, pp. 117–124.

[15] N. R. Pal, J. C. Bezdek, and E. C. Tsao, "Generalized clustering networks and Kononen's self-organizing scheme," *IEEE Trans. Neural Net.,* vol. 4, no. 4, pp. 549–557, 1993.

[16] T. N. Yang and S. D. Wang, " A generalized competitive clustering algorithm," in *Int. Symp. on Artificial Neural Net.*, 1994, pp. 365–373.

[17] A. I. Gonzalez, M. Grana, and A. D'Anjou, "An analysis of the GLVQ algorithm," *IEEE Trans. Neural Net.,* vol. 6, pp. 1012–1016, 1995.

[18] N. R. Pal, J. C. Bezdek, and E. C. Tsao, "Repair to GLVQ: a new family of Competitive Learning Schemes," *IEEE Trans. Neural Net.,* vol. 7, pp. 1062–1070, 1996.

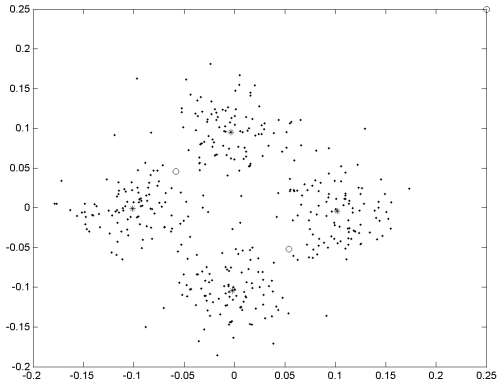

Figure 3: Final positions of LVQ prototypes. Actual centroids: '*'. LVQ prototypes: 'o'.

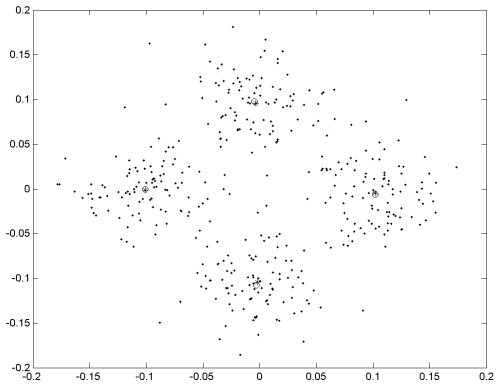

Figure 4: Final positions of HRC prototypes. Actual centroids: '*'. HRC prototypes: 'o'.

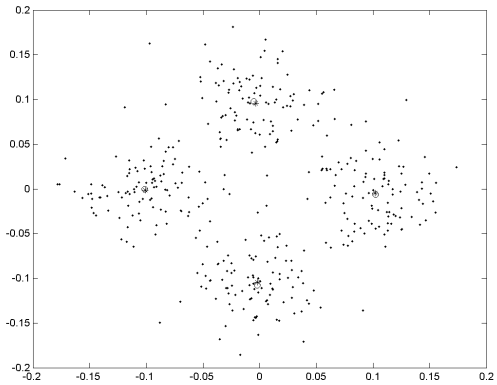

Figure 5: FSCL prototypes when the number of data in each cluster is equal.

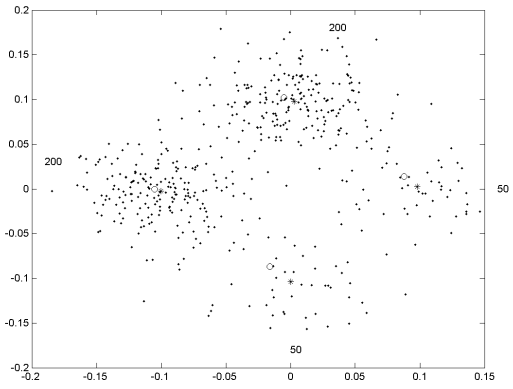

Figure 6: FSCL prototypes when the number of data in each cluster is not equal.

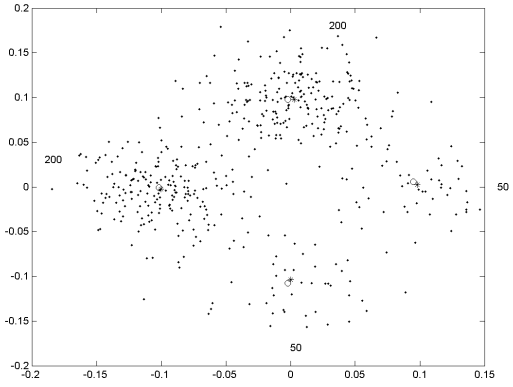

Figure 7: HRC prototypes when the number of data in each cluster is not equal.

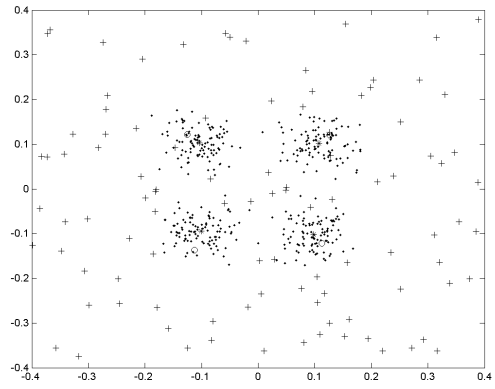

Figure 8: Final positions of HRC prototypes when outliers exist. Actual centroids: '*'. HRC prototypes: 'o'.

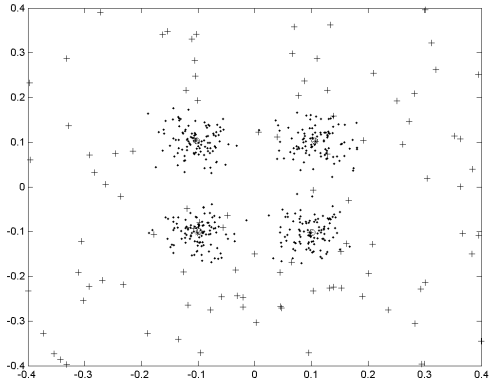

Figure 9: Final positions of FRC prototypes when outliers exist.

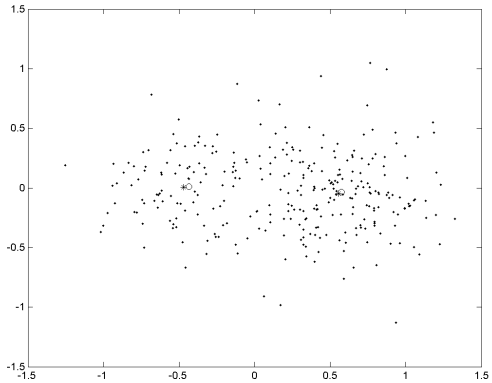

Figure 10: Final positions of FRC prototypes when two overlapped clusters exist.

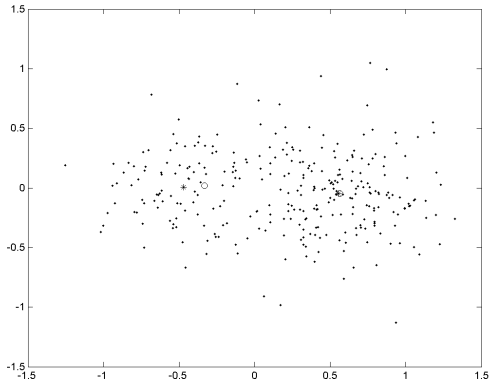

Figure 11: Final positions of PCM prototypes when two overlapped clusters exist.