Fuzzy Robust Clustering and C Spherical Shells Algorithms

Tai-Ning, Yang
Department of Computer Science
Chinese Culture University
tnyang@faculty.pccu.edu.tw

Sheng-De, Wang
Department of Electrical Engineering
National Taiwan University
sdwang@cc.ee.ntu.edu.tw

Abstract

Clustering algorithms play an important role in the field of pattern recognition. Most of the traditional clustering algorithms are hard clustering. Some algorithms based on fuzzy set theorem have been proposed recently. Some algorithms based on the fuzzy clustering theory have been proposed recently. They modify not only the winner but also other prototypes for each input. Many fuzzy clustering algorithms including GLVQ-F use the membership from fuzzy c-means (FCM). Since the objective function in FCM is constrained by a probability premise, the total sum of the membership shared by all classes for an input data must be one. If the number of classes is large, the fuzzy membership may be minute. It has been shown that if the number of classes is large, the traditional clustering algorithms may be better than the fuzzy ones. Moreover, the outliers that may appear are not considered in FCM. We propose a new family of clustering algorithms called fuzzy robust clustering (FRC) without the above disadvantages. The size of updating prototype is independent of the number of prototypes and the influence of outliers is reduced. We modify the objective function and propose a robust algorithm for the extraction of spherical shells. Artificially generated data are used to test FRC.

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 related books [1], [2], [3].

Recently the on-line competitive learning in neural networks is often associated with the sequential partitional clustering [4]. The searching and updating process for an appropriate prototype is considered as the competition between the hidden neurons. It was pointed out that the basic form of the competitive learning is the same as the sequential version of hard c-means. Kohonen [5], [6] proposed a competitive algorithm called learning vector quantization (LVQ). LVQ attracts a lot of attentions because of its simplicity and efficiency. There are many variants of LVQ. For example, Kohonen proposed two refinements, call LVQ2 and LVQ3. Unfortunately LVQ suffers serious prototype under-utilization problem.

The other parts of this paper are organized as follows. In section 2, we review generalized learning vector quantization-fixed (GLVQ-F), fuzzy algorithms for learning vector quantization (FALVQ) and fuzzy c means (FCM). The following section introduces our algorithm called fuzzy robust clustering (FRC). Fuzzy robust c spherical shells (FCSS) is propoesd in section 4. Section 5 contains the simulations. Section 6 is the summary.

2 Fuzzy clustering algorithms

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 $1 \leq i \leq c$.

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). This situation is also analyzed by Gonzalez et al. [7] Another modified algorithm called GLVQ-F is designed by Nikhil [8] et al., where a new loss function is pro-

posed:

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

where $w_{ij} = (\sum_{r=1}^{c} \frac{\|x_j - v_i\|^{2/(m-1)}}{\|x_j - v_r\|^{2/(m-1)}})^{-1}$. w_{ij} is the membership from fuzzy c-means (FCM). Using the similar gradient descent approach as in GLVQ, they derived the following algorithm. GLVQ-F algorithm: the same as Competitive Learning except(CL) the following updating formula.

Update each prototype:

$$v_i^{new} = v_i^{old} \qquad (2)$$

$$+(\frac{\alpha_t}{m-1})(x_k - v_i^{old})(m-2)w_{ij} \qquad (3)$$

$$+(\sum_{r=1}^{c} \frac{\|x_k - v_i\|^{2/(m-1)}}{\|x_k - v_r\|^{2/(m-1)}})^{(2-m)} w_{ij}^2). \qquad (4)$$

The size of updating prototypes in GLVQ and GLVQ-F is affected by c, the number of clusters. According to the analysis by Gonzalez et al. [7], the performance of GLVQ type algorithms may considerably depend on the size of c. Initialization should be careful when using GLVQ-F. Since the prototypes located at the same place updated in the same way, different prototypes must be set at different locations. From (2), the distance between the input and prototypes can't be zero. So it is better not to use a strategy to initialize the prototypes with the input data. There are other fuzzy clustering algorithms using the membership from FCM, such as FLVQ [10], [11]. Karayiannis and Pai [12] proposed another form of objective function: They propose a new loss function:

$$E(V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n} w_{ij}\|x_j - v_i\|^2, \qquad (5)$$

where $w_{ij} = 1$ if v_i is the winner else $w_{rj} = u(\frac{\|x_j - v_i\|^2}{\|x_j - v_r\|^2})$. The derived algorithms are called *Fuzzy Algorithms for Learning Vector Quantization* (FALVQ) [12]. Different selections of weighting function u lead to FALVQ1, FALVQ2, and FALVQ3. Because the distance between current input and its corresponding winner is a variable of the weighting function for the losers, the adjustments are strongly affected by the distance between the winner and current input. Moreover the derived winner updating formula also includes an influence term from the losers.

We will discuss the inappropriate influence of the cluster number from the analysis of fuzzy c-means [9]. Let us use U to denote the membership matrix. The objective function in *Fuzzy C-Means* (FCM) is:

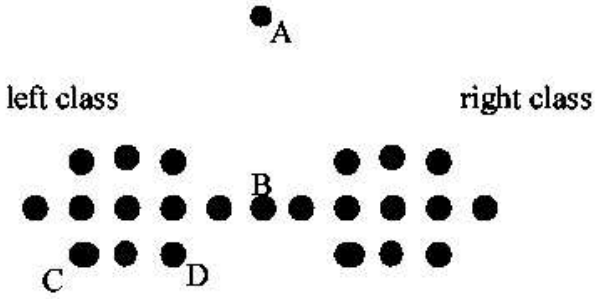

Figure 1: Two clusters and one outlier.

$$E(V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n} {u_{ij}}^m \|x_j - v_i\|^2. \qquad (6)$$

In the above equation, $m \in [1,\infty)$ is a weighting 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.

Bezdek [9] applied the technique of Lagrange multiplier and derived the optimal membership function,

$$u_{ij} = (\sum_{r=1}^{c} \frac{\|x_j - v_i\|^{2/(m-1)}}{\|x_j - v_r\|^{2/(m-1)}})^{-1}. \qquad (7)$$

A simple example in Fig. 1 illustrates the problems caused by the probabilistic constraint. In the FCM type algorithm, point A and point B may have the same membership value. But apparently point A is an outlier. Although point C and point D are symmetric to the left cluster, their memberships are different. Probabilistic constraint forces the total sum of the membership for each input data must be one, so the outlier could not be distinguished. Since the total membership is shared by all classes, this forces point D to transfer some left class membership value to right class. If the number of classes c is large, the fuzzy membership may be minute. Moreover, this objective function does not consider the situation when outliers exist.

3 Fuzzy robust clustering algorithms

If we simply relax the constraint 3 without modifying (6), a trivial solution in which all member-

ships are zero will be produced. A simple modification may be like this

$$E(V,X)=\sum_{i=1}^{c}\sum_{j=1}^{n}{u_{ij}}^{m}\|x_j-v_i\|^2+ \tag{8}$$

$$\sum_{i=1}^{c}\sum_{j=1}^{n}(1-u_{ij})^{m}\|x_j-v_i\|^2. \tag{9}$$

Again, another trivial solution in which all memberships are $\frac{1}{2}$ is derived.

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 j-th data cluster. Thus the fuzzy complement is the membership of x_j in the noise cluster with the constant distance η_i. The general form of the proposed objective function is:

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

$$+\sum_{i=1}^{c}\sum_{j=1}^{n}(f(u_{ij}))^{m}\eta_i. \tag{11}$$

$d(x_j,v_i)$ is the distance measure from the data point x_j to the prototype v_i. For the simplicity of the following discussion, we set $m=1$ and $f(u_{ij})=1+u_{ij}*log(u_{ij})-u_{ij}$. It is easy to prove that the equation, $1+u_{ij}*log(u_{ij})-u_{ij}$, satisfies the axioms of fuzzy complements, boundary conditions and monotonicity [14]. Other fuzzy complements may also be used. The following is the proposed fuzzy objective function:

$$FE(V,X)=\sum_{i=1}^{c}\sum_{j=1}^{n}u_{ij}d^2(x_j,v_i) \tag{12}$$

$$+\sum_{i=1}^{c}\sum_{j=1}^{n}(1+u_{ij}*log(u_{ij})-u_{ij})\eta_i. \tag{13}$$

In the clustering, 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

$$u_{ij}=exp(-\frac{\|x_j-v_i\|^2}{\eta_i}). \tag{14}$$

Substitute this membership back and after simplification, we get

$$FE(V,X)=\sum_{i=1}^{c}\eta_i-\sum_{i=1}^{c}\sum_{j=1}^{n}exp(-\frac{\|x_j-v_i\|^2}{\eta_i})\eta_i. \tag{15}$$

Following the multidimensional chain rule, when x_j is the current input data the gradient of FE with respect to v_i is

$$\frac{n\partial FE(V,X)}{\partial v_i} \tag{16}$$

$$=\Big(\frac{n\partial FE}{\partial\|x_j-v_i\|^2}\Big)\Big(\frac{\partial\|x_j-v_i\|^2}{\partial v_i}\Big) \tag{17}$$

$$=u_{ij}\Big(\frac{\partial\|x_j-v_i\|^2}{\partial v_i}\Big) \tag{18}$$

$$=-2u_{ij}(x_j-v_i). \tag{19}$$

In (14), η_i plays the role of a normalization parameter for the distance $\|x_j-v_i\|^2$ in the membership u_{ij}. In Fig. 2, we show the membership relative to $\eta_i=1,2,..10$, respectively. Note that $exp(-\|x_j-v_i\|^2/1) < exp(-\|x_j-v_i\|^2/2) < ... < exp(-\|x_j-v_i\|^2/10)$.

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 (14) 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.

Following the similar approach, we have developed a neural network for the extraction of principal components in the noisy data set. [15]

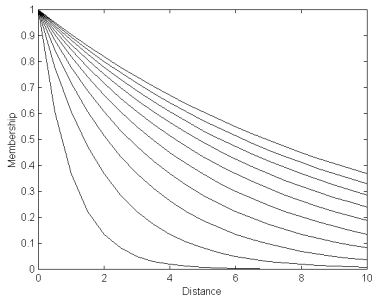

Figure 2: Plot of the membership generated with different η.

4 Fuzzy robust c spherical shells algorithm

The fuzzy c spherical shells algorithm is designed for the searching of the clusters belonging to the spherical shells. Raghu Krishnapuram et al. [13] derived *fuzzy c spherical shells* (FCSS) algorithm with the membership of fuzzy c-means (7). We define the prototype $v_i = (o_i, r_i)$ and set $d^2(x_j, v_i) = (\|x_j - o_i\|^2 - r_i^2)^2$, where o_i is the center of the $i-th$ hypersphere and r_i is the radius. Following (14), the membership weighted loss function is:

$$FE(V,X) = \sum_{i=1}^{c}\sum_{j=1}^{n} exp(-\frac{(\|x_j - o_i\|^2 - r_i^2)^2}{\eta_i}) \quad (20)$$

$$(\|x_j - o_i\|^2 - r_i^2)^2. \quad (21)$$

We rewrite $d^2(x_j, v_i) = p_i^t M_j p_i + y_j^t p_i + b_j$, where $b_j = (x_j^t x_j)^2$, $y_j = 2(x_j^t x_j)\begin{pmatrix} x_j \\ 1 \end{pmatrix}$, $p_i = \begin{pmatrix} -2o_i \\ o_i^t o_i - r_i^2 \end{pmatrix}$, and $M_j = \begin{pmatrix} x_j \\ 1 \end{pmatrix}\begin{pmatrix} x_j \\ 1 \end{pmatrix}^t$. It has been shown in [13], the updating of p_i in the membership weighted form is

$$p_i = -(\frac{1}{2})(H_i)^{-1} w_i, \quad (22)$$

where

$$H_i = \sum_{j=1}^{n} u_{ij} M_j \quad (23)$$

and

$$w_i = \sum_{j=1}^{n} u_{ij} y_j. \quad (24)$$

Fuzzy Robust C Spherical Shells (FRCSS) algorithm:

Input: all of the training data 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.

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

step 3. Compute H_i and w_i for each cluster with (23) and (24). Compute p_i for each cluster with (22).

step 4. Compute u_{ij} with (14).

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 CL and FRC

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 $T = 40$, $c = 4$ and $\alpha_0 = 0.2$. Both strategies for adjusting η_i produce the same results.

Results: As shown in Fig. 3, the final positions of prototypes in CL are greatly affected by the outliers. The final prototypes of FRC are near the actual centroids as shown in Fig. 4.

5.2 Extraction of 2-D circles

Input data: There are three 2-D circles and each circle has 100 boundary points marked by dot. There are 100 outliers marked by "+".

Initialization: Initial positions of prototypes in FCSS [13] are randomly set and the final prototypes are initial prototypes of FRCSS. We set $T = 40$, $c = 4$ and $\alpha_0 = 0.2$.

Results: As shown in Fig. 5, the extracted circles with FCSS are greatly affected by the outliers. The extracted circles with FRCSS are near the actual circles as shown in Fig. 6.

5.3 Extraction of 3-D spherical shells

Input data: There are three 3-D spheres and each sphere has 100 boundary points marked by dot. There are 100 outliers marked by "+".

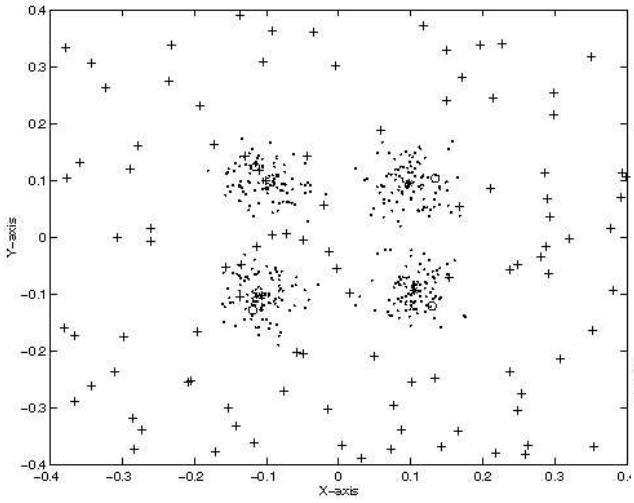

Figure 3: Final positions of CL prototypes when outliers exist. Actual centroids: '*'. CL prototypes: 'o'.

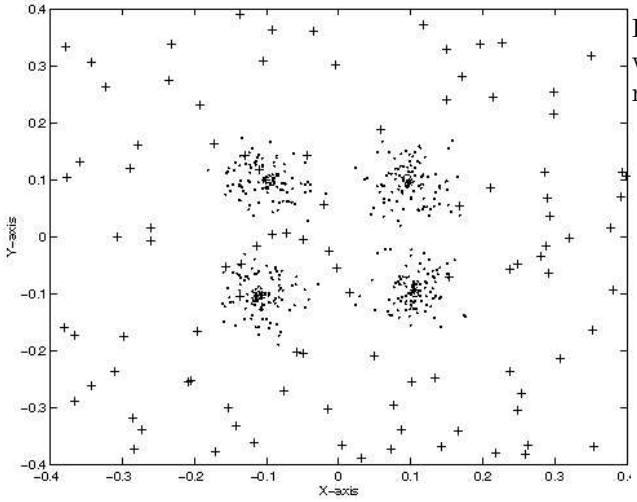

Figure 4: Final positions of FRC prototypes when outliers exist.

Initialization: Initial positions of prototypes in FCSS are randomly set and the final prototypes are initial prototypes of FRCSS. We set $T = 40$, $c = 4$ and $\alpha_0 = 0.2$.

Results: As shown in Fig. 7, the extracted spherical shells with FCSS are greatly affected by the outliers. The extracted spherical shells with FRCSS are near the actual circles as shown in Fig. 8.

6 Summary

Through a modification of the objective function with the releasing of probabilistic constraints, we derive a family of robust clustering algorithms and

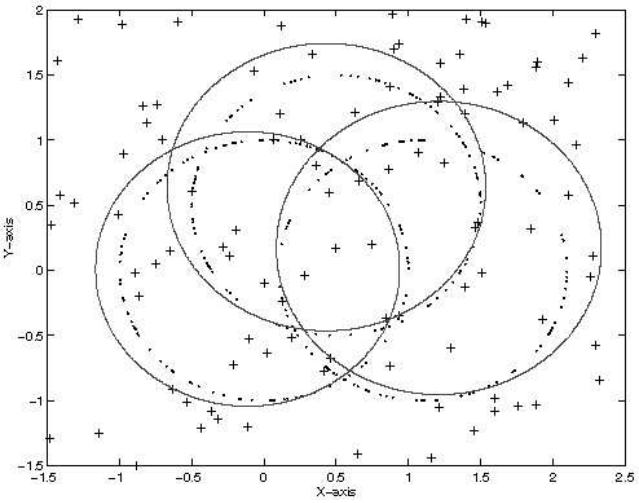

Figure 5: Final positions of FCSS prototypes when outliers exist and the 2-D testing data. "+" represents the outlier.

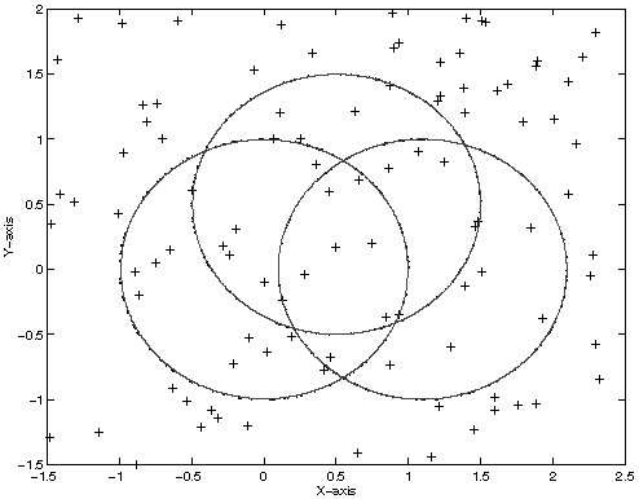

Figure 6: Final positions of FRCSS prototypes when outliers exist and the 2-D testing data. "+" represents the outlier.

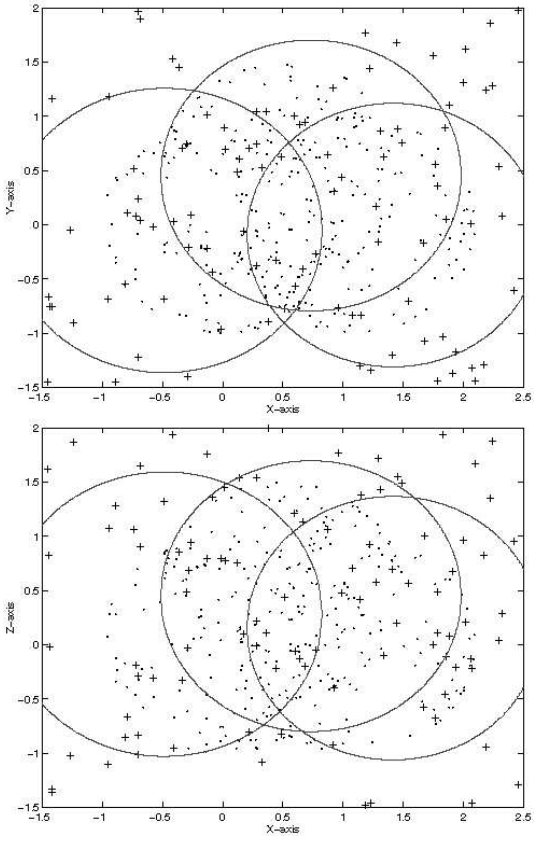

Figure 7: Final positions of FCSS prototypes when outliers exist and the projection of the 3-D testing data on the x-y and x-z plane. "+" represents the outlier.

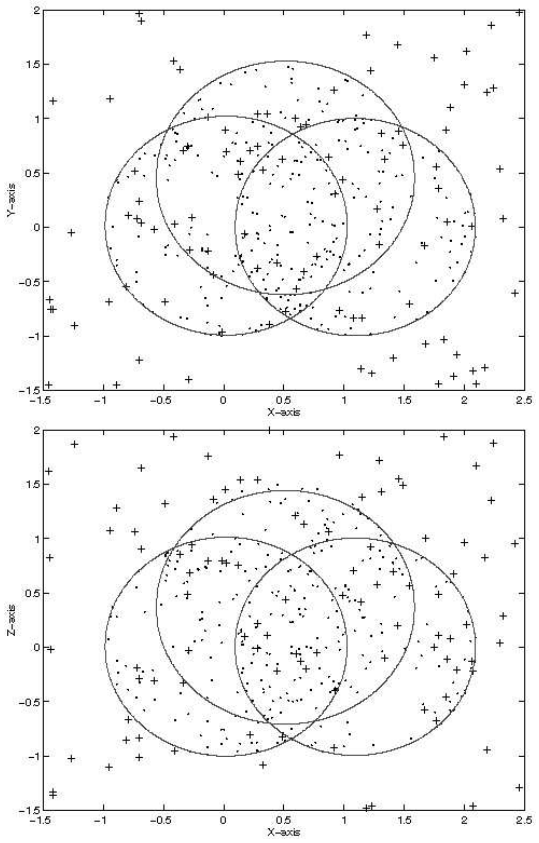

Figure 8: Final positions of FRCSS prototypes when outliers exist and the projection of the 3-D testing data on the x-y and x-z plane.

the corresponding fuzzy c spherical shells algorithms. Compared with the previous hard and fuzzy clustering, FRC and FRCSS have the following distinctive features:

- the membership does not reduce with the growth of the cluster number.

- it is not necessary to find the winner.

- alleviating the influence from the outlier.

The normalized distance between the current input and each prototype decides the adjustment of this prototype. As supported by the experiments, FRC and FRCSS are effective and could be used to improve results from another clustering algorithm. There exists other form of FRC-like algorithms. One simple modification is to change the learning law to batch mode or using a momentum updating law. These alterations may be better than the original algorithm if the input presentation order is biased. Other strategies to estimate the zone of influence may be possible for better adjustment of the learning rate.

References

[1] P. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach.* Englewood Cliffs, NJ: Prentice-Hall, 1982.

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

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

[4] J. Buhmann and H. Kuhnel, "Complexity optimized data clustering by competitive neural networks," *Neural Computation,* vol. 5, pp. 75–88, 1993.

[5] T. Kohonen, *Self-organization and associative memory.* Berlin, Germany : Springer-Verlag, 3rd ed. 1989.

[6] T. Kohonen, "The self-organizing map," in *Proc. IEEE,* vol. 78, 1990, pp. 1464–1480.

[7] 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.

[8] 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.

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

[10] F. L. Chung and T. Lee, "Fuzzy competitive learning," *Neural Networks,* vol. 7, no. 3, pp. 539–551, 1994.

[11] J. C. Bezdek and N. R. Pal, "Two soft relatives of vector quantization," *Neural Networks,* vol. 8, no. 5, pp. 729–743, 1995.

[12] N. B. Karayiannis and P. I. Pai, "Fuzzy algorithms for learning vector quantization," *IEEE Trans. Neural Net.,* vol. 7, pp. 1196–1211, 1996.

[13] 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.

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

[15] Tai-Ning, Yang and Sheng-De, Wang, "Fuzzy Auto-associative Neural Network for Principal Component Extraction of Noisy Data," *IEEE Trans. Neural Net.,* vol. 11, no. 3, pp. 808–810, 2000.