# Branching Competitive Learning for Clustering

Irwin King

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong
Email: king@cse.cuhk.edu.hk

Huilin Xiong

Department of Mathematics
Huazhong University of Science and Technology
Wuhan, Hubei, China
Email: hlxiong@cse.cuhk.edu.hk

## Abstract

*This paper presents a novel modification to the classical Competitive Learning (CL) by adding a Branching and Merging Criteria so that the estimated number of cluster centers can increase over time until it reaches a good approximation of the number of data clusters. This technique shows a faster convergence to the approximate cluster centers, and more importantly, shows the ability to estimate the number of clusters in the input data distribution. We illustrate the formulation of the branching criteria and demonstrate the efficiency of our Branching Competitive Learning (BCL) for data clustering through a set of experiments.*

## 1 Introduction

Competitive Learning (CL) for data clustering, as an adaptive version of the classical K-mean clustering, has many applications in the fields such as clustering analysis, pattern recognition [1], and image coding based on vector quantization [5]. Recently, CL and its modification RPCL (Rival Penalized Competitive Leaning) [7, 6] were used by some researchers [3, 4] in the applications of image database indexing to speed up image information retrieval.

One of the key problems in most clustering algorithms whether it is CL and K-mean is that the number of clusters must be appropriately preselected; otherwise, these clustering algorithms may have poor performance. For example, one may over- or under-estimate the number of clusters and this can result in unfavorable outputs. By now, how to correctly and efficiently estimate the number of clusters in a data distribution is still an open problem, although this can be tackled in a sense by some heuristic techniques, e.g., ISODATA [2], or by RPCL [7].

In RPCL, a "push-pull" mechanism is adopted to drive the correct number of moving centers (or synaptic weight vectors of neural units) towards the relevant cluster centers. The "pull" mechanism is similar to the mechanism found in CL which draws the winner moving to a randomly selected input data point. On the other hand, the "push" mechanism repels the second winner (or the rival) away from the randomly selected input data point. This procedure can be expressed by:

$$\begin{aligned}
\vec{w}_c(t+1) &= \vec{w}_c(t) + \alpha_c(\vec{x} - \vec{w}_c(t)) \\
&\quad \text{for } c = \text{argmin}_j \gamma_j \|\vec{x} - \vec{w}_j\|^2 \\
\vec{w}_r(t+1) &= \vec{w}_r(t) + \alpha_r(\vec{x} - \vec{w}_r(t)) \\
&\quad \text{for } r = \text{argmin}_{j \neq c} \gamma_j \|\vec{x} - \vec{w}_j\|^2 \\
\vec{w}_j(t+1) &= \vec{w}_j(t) \\
&\quad \text{for } j \neq c, r
\end{aligned} \tag{1}$$

where $\vec{x}$ is a randomly selected input data point, $t$ represents the current step of competitive learning, $\alpha_c$ and $\alpha_r$ are the learning rates for the push and pull factor respectively (usually $\alpha_c \gg \alpha_r$), and $\gamma_j$ is the frequency that $\vec{w}_j$ wins the competition up to now.

Many experiments have shown that RPCL can work well provided that the parameters are properly tuned beforehand. Experimentally, the parameter tuning in RPCL depends on the data distribution and the algorithm is sensitive to the preselected number of clusters and the initialization of the weight vectors.

In this paper, we present a novel modification to the classical Competitive Learning by adding a branching mechanism where a new center (or weight vector) can be spawned off from a previous center. Our clustering scheme starts from one center (seed), and then this seed will split to form other new centers during the process of competitive learning. In this way, the number of branching centers can be increased to approximate the actual number of clusters in input data to give a good clustering result. Experimentally, this Branching Competitive Learning (BCL) scheme shows a faster convergence of weight vectors to the approximate cluster centers, and more importantly, it demonstrates the ability to estimate the number of clusters in the input data distribution. We will illustrate the formulation of the branching criteria and demonstrate the novel ability of BCL through a set of experiments.

Figure 1: An illustration of the procedure of the BCL algorithm when the inital seed is branched into three centers to accommodate the three clusters.

In the next section, we will detail the branching criteria. The experiments can be found in Section 3. We will then make some final remarks in the conclusion section.

## 2 Branching Competitive Learning

In this section, we present a branching criteria in competitive learning and detail the algorithm of our Branching Competitive Learning for clustering.

### 2.1 Branching Criteria

Given a data distribution which has two clusters, if the preselected cluster number in the classical CL algorithm is set to one, it is obvious that the unique moving center will display an oscillating behavior when it comes near the two data clusters and the procedure of competitive learning cannot converge. Intuitively, if a new center can be spawned off from the original one somewhere along its trajectory, the procedure of the competitive learning will converge faster and a good clustering result can be given by CL algorithm. The situation for more clusters is similar to this, which can be illustrated by Fig. 1.

In BCL, there are two conditions for a seed or a moving center to branch off a new center. The first is the *Angle Criterion* which is based on the angle between its current moving direction and the previous one of the center. The second is the *Distance Criterion* which is based on the distance between the input data $\vec{x}$ and winner $\vec{w_c}$:

$$\begin{aligned} \angle\langle \vec{x}(t) - \vec{w}_c, \vec{x}(t-1) - \vec{w}_c \rangle &> \varphi_0 \\ \|\vec{x}(t) - \vec{w}_c\| \cdot \|\vec{x}(t-1) - \vec{w}_c\| &> d_0 \end{aligned} \quad (2)$$

where $\varphi_0$ and $d_0$ are angle and distance criteria thresholds to control the branching process respectively.

We call the above conditions Branching Criteria. Usually, the threshold $\varphi_0$ in the first inequation of the branching criterion is set to 90°, and therefore, the first condition of Eq. (2) becomes

$$(\vec{x}(t) - \vec{w}_c) \cdot (\vec{x}(t-1) - \vec{w}_c) < 0. \quad (3)$$

The second condition of the branching criteria (2) can be viewed as a stopping criteria for centers branching, because when a moving center enters the interior of a cluster, this condition cannot be met anymore, and therefore, the moving center will just move toward the center of the cluster and cannot split again.

### 2.2 The Algorithm of Branching Competitive Learning

Combining the branching criteria (2) and classical competitive learning, we formulate the algorithm of our Branching Competitive Learning below:

1. Initialize the "seed" or the first moving center.

2. Randomly select a sample $\vec{x}$ from data set, find the winner $\vec{w}_c$ of current competition in the set of moving centers $\{\vec{w}_j\}$ $(j = 1, 2, \cdots, n)$, that is $c = \text{argmin}_j \gamma_j \|\vec{x} - \vec{w}_j\|^2$, where $\gamma_j$ is the frequency that $\vec{w}_j$ wins the competition up to now.

3. if $\vec{w}_c$ satisfies the branching criteria (2), a new moving center $\vec{w}_{n+1}$ is spawn off from $\vec{w}_c$:

$$\vec{w}_{n+1} = \vec{w}_c + \alpha_c(\vec{x} - \vec{w}_c)$$

otherwise, update $\vec{w}_c$ by $\vec{w}_c + \alpha_c(\vec{x} - \vec{w}_c)$.

In BCL, because there is only one moving center at the start, the so called *dead unit problem* in classical CL will not appear. Moreover, as there is no other rivals at first, the moving center approaches the data set faster than CL and RPCL, and when it comes near data cluster after some splittings it will moves slowly to approximate the center of data cluster. In the next section, we will demonstrate this through a sets of experiments.

## 3 Experiments

To test the efficiency of our Branching Competitive Learning for the task of estimating cluster number and clustering data, we conducted three sets of experiments. The first set of experiments will examine the ability of our BCL to detect cluster number. The second set of experiments will show a natural way to implement a hierarchical or a multiresolution clustering in BCL scheme. The third set of experiments will compare the performances of BCL and RPCL in terms of accuracy and speed for data clustering.

Our experimental environment consists of a Pentium II PC with 128 Meg of internal memory running on the Windows98 operating system. We implemented the BCL and other related algorithms for our experiments in Visual C++ 6.0.

### 3.1 Cluster Number Detection by BCL

In the first set of experiments, we use three 2-dimensional data sets to test the validity of the branching criteria for the task of cluster number detection. Each data set has 1,000 samples. We fix the learning rate at $\alpha_c = 0.05$.

Date set 1, as shown in Fig. 2, contains four clusters of Gaussian distribution with $\sigma = 0.5$ and centered at $(-3, 0)$, $(3, 0)$, $(0, -3)$, and $(0, 3)$ respectively. The original moving center (or the seed) is initialized by random numbers in the interval $[5.8, 6.0]$ (Fig. 2(a)), or by random numbers in the interval $[-6.0, -5.8]$ (Fig. 2(b)), or by the point (0,0) (Fig. 2(c)). The threshold of angle and distance in branching criteria (2) are set with $\varphi_0 = 90°$, $d_0 = 6.0$, and convergence tolerance is set with $\varepsilon = 0.05$. Fig. 2 shows the traces of moving centers and their branching status.

Date set 2, as shown in Fig. 3, is similar to data set 1, but the centers of the four clusters of Gaussian distribution are located at $(-2, 0)$, $(2, 0)$, $(0, -2)$, and $(0, 2)$ respectively. Obviously, the clusters in data set 2 are closer to each other than in data set 1. The original moving center is initialized by random numbers in the interval $[3.8, 4.0]$ (Fig. 3(a)), or by random numbers in the interval $[-4.0, -3.8]$ (Fig. 3(b)), or by the point (0,0) (Fig. 3(c)). The threshold of angle and distance in branching criteria (2) are set with $\varphi_0 = 90°$, $d_0 = 4.0$, and convergence tolerance is set to 0.04. Fig. 3 shows the traces of moving centers and their branching status.

In date set 3, as shown in Fig. 4, the centers of the four clusters of Gaussian distribution are located at $(-1.5, 0)$, $(1.5, 0)$, $(0, -1.5)$, and $(0, 1.5)$ respectively. Clearly, there are some overlapping between clusters of data set 3. The original moving center is initialized by random numbers in the interval $[2.8, 3.0]$ (Fig. 4(a)), or by random numbers in the interval $[-3.0, -2.8]$ (Fig. 4(b)), or by the point (0,0) (Fig. 4(c)). The threshold of angle and distance in branching criteria (2) are set with $\varphi_0 = 90°$, $d_0 = 3.3 \sim 3.5$, and convergence tolerance is set as data set 2.

From the traces of the moving centers, we can see that BCL can automatically and correctly detect the number of clusters in data distribution even for the situation that there are some overlapping between clusters (Fig. (4)). In the experiments, we also found that when the clusters have some overlapping between them, the parameters $d_0$ and the convergence tolerance $\varepsilon$ should be finely tuned to obtain a good result.

### 3.2 Multiresolution clustering

The threshold parameter $d_0$ plays a key role in controling the branching process. A large value setting would result in an under estimation of cluster number. From a viewpoint of multiresolution or hierarchy, this phenomena is natural, because when we observe the data distributions in a view of large scale, an aggregation of many different clusters in small scales could be view as just one cluster. So, in a sense, the threshold $d_0$ acts as a resolution control on how we view the data distribution. By using it, we can conveniently implement a multiresolution or hierachical data clustering.

Here we use a 2-dimension data set, shown in Fig. 5, to illustrate a multiscale BCL for clustering. The data set consists of 16 Gaussian distributions with standard variance ($\sigma$) 0.8. Each cluster has 200 samples, and the total number of sample is 3200. Obviously, in a view of relative large scale, there are just four clusters in the data set. However, if we view the data set in a smaller scale, we can find 16 clusters in it.

In this experiment, we conduct a 2-level multiresolution clustering by BCL scheme. In level 1, we set the distance threshold of Branching Criteria 2 to be $d_1 = 10.0$, which corresponds to the resolution of level 1. In leve 2, the distance threshold is set to be $d_2 = 2.8$. The learning rate is fixed at $\alpha = 0.02$.

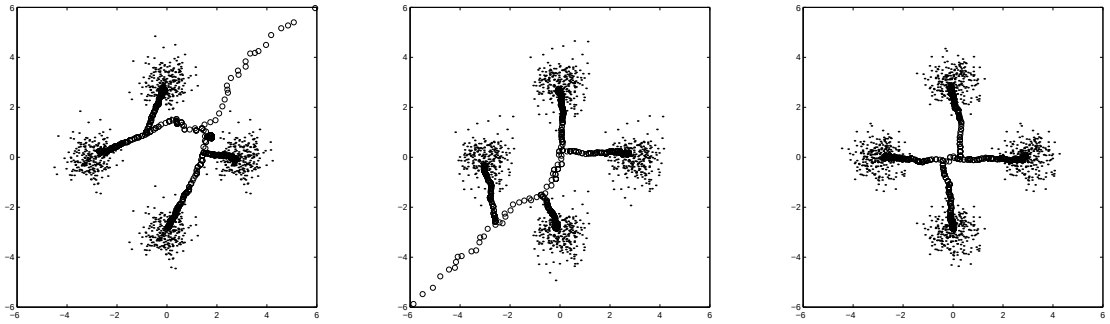
The experimental results are given in Fig. 5, Fig. 5 (a) shows the original data set. Fig. 5 (b) shows the learning traces of moving centers in level 1. From it, we can see that in the resolution level 1, data points are dived into four clusters and the convergent moving centers are correctly located at the centers of each data clusters. Fig. 5 (c) shows the learning traces of the further branched moving centers in resolution level 2, and it can be seen that the finally detected cluster number is right the correct number of clusters. Fig. 5 (d) gives the positions of the cluster centers and the final estimated centers. It can be seen that the deviations between them are very small.

### 3.3 Comparison of BCL and RPCL

We now compare the performance of CL and RPCL algorithms for data clustering. There are two metrics we used for comparing them. One is the accuracy of clustering, which is measured by the average percentage of correct classification of data, and the other is the speed or the running time of the algorithms.

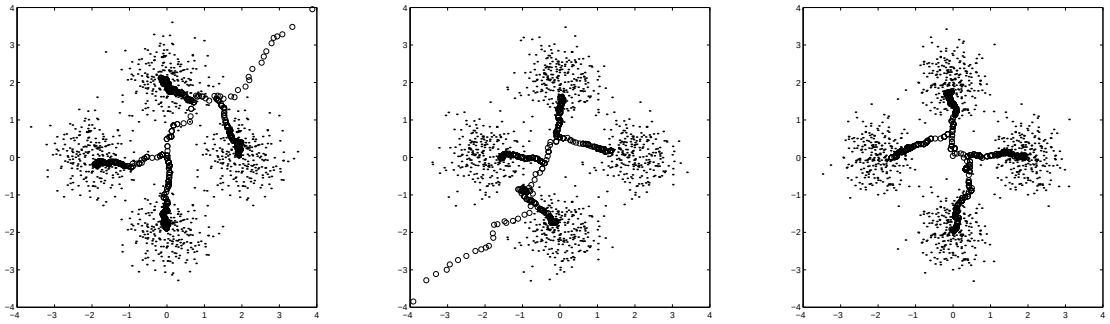
The data used in this set of experiments are 5-dimensional points. There are four data sets, and each data set has ten clusters of samples which come from Gaussian distributions with standard variance 0.5 or from uniform distributions in 5-dimension cubes. The total number of samples in each data set is 2,000, and each cluster contains 200 samples. These data sets are listed below:

- Data set 4: this data set has 2,000 samples from 5-dimensional uniform distributions. The centers of the ten clusters are located at ($\pm$1.5, 0, 0, 0, 0), (0, $\pm$1.5, 0, 0, 0), (0, 0, $\pm$1.5, 0,



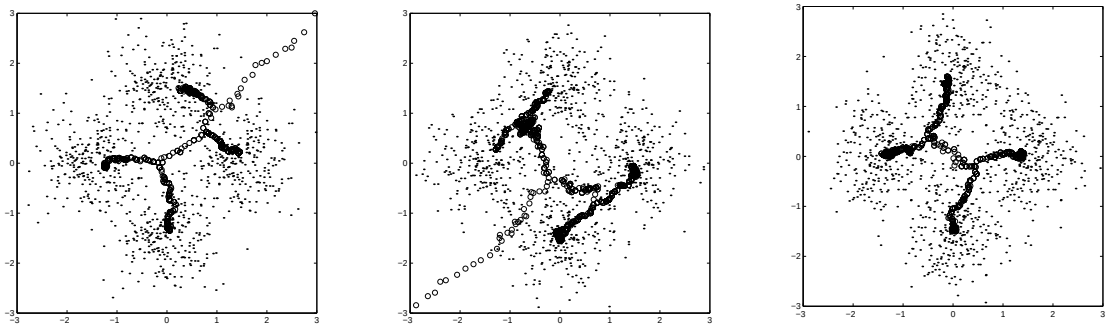
(a) (b) (c)

Figure 2: Data set 1 and the branching traces of moving centers.



(a) (b) (c)

Figure 3: Data set 2 and the branching traces of moving centers.



(a) (b) (c)

Figure 4: Data set 3 and the branching traces of moving centers.

| | ave. correct | speed (s) |
|---|---|---|
| data set 4 | 94.00 % | 97.29 |
| data set 5 | 94.74 % | 68.65 |
| data set 6 | 94.88 % | 82.89 |
| data set 7 | 97.92 % | 71.91 |

Table 1: The average percentage of correct classification and speed of RPCL.

| | ave. correct | speed (s) |
|---|---|---|
| data set 4 | 100 % | 31.02 |
| data set 5 | 100 % | 40.96 |
| data set 6 | 96.27 % | 62.71 |
| data set 7 | 97.16 % | 50.61 |

Table 2: The average percentage of correct classification and speed of BCL

0), (0, 0, 0, ±1.5, 0), and (0, 0, 0, 0, ±1.5) respectively.

- Data set 5: this data set has 2,000 samples from 5-dimensional uniform distributions. The centers of the ten clusters are located at (±2, 0, 0, 0, 0), (0, ±2, 0, 0, 0), (0, 0, ±2, 0, 0), (0, 0, 0, ±2, 0), and (0, 0, 0, 0, ±2) respectively.

- Data set 6: Data set 4: this data set has 2,000 samples from 5-dimensional Gaussian distributions. The centers of the Gaussian distributions are (±1.5, 0, 0, 0, 0), (0, ±1.5, 0, 0, 0), (0, 0, ±1.5, 0, 0), (0, 0, 0, ±1.5, 0), and (0, 0, 0, 0, ±1.5) respectively.

- Data set 7: Data set 4: this data set also has 2,000 samples from 5-dimensional Gaussian distributions. The centers of the Gaussian distributions are (±2, 0, 0, 0, 0), (0, ±2, 0, 0, 0), (0, 0, ±2, 0, 0), (0, 0, 0, ±2, 0), and (0, 0, 0, 0, ±2) respectively.

We set the convergence tolerance with $\varepsilon = 0.04$, and the learning rate $\alpha_c$ to 0.05 for RPCL and BCL. For RPCL, the rival's learning rate was set from 0.002 to 0.004. For BCL, the threshold $d_0$ was set from 2.5 to 7.5 for uniform distributions, and from 3.8 to 8.0 for Gaussian distributions respectively.

Table (1) and Table (2) show the average clustering accuracy and speed of RPCL and BCL on data sets above. The values in each table were calculated from ten consecutive runs of the algorithm. It can be seen that BCL converges faster than RPCL, and, for uniform distributions, BCL gives more accurate clustering than RPCL. However, for Gaussian distributions, two of the experiments show almost the same clustering accuracy. From our experiments, we observed that the BCL algorithm has a few good qualities that can be used in some image indexing applications. We plan to investigate further how the algorithm would perform under various data distributions, initial conditions, and various thresholding techniques to maximize the BCL algorithm.

## 4 Conclusion

We present a novel modification to the classical Competitive Learning by adding a Branching Criteria so that the number of moving centers can increase over time until it reaches a good approximation of the number of data clusters. This technique shows a faster convergence to the approximate cluster centers, and more importantly, it is capable of detecting cluster number in the input data distribution.

## References


[1] P.A. Devijiver and J. Kittler. *Pattern Recognition - A Statistical Approach*. Pretice-Hall, 1982.

[2] A.K. Jain and R.C. Dubes. *Algorithm for clustering Data*. Prentice-hall, 1988.

[3] I. King and T.K. Lau. Comparison of several partitioning methods for information retrieval in image database. *Proceedings of the 1997 International Symposium on Multimedia Information Processing (ISMIP'97)*, 1997.

[4] I. King, L. Xu, and L.W. Chan. Using rival penalized competitive clustering for feature indexing in hong kong's textile and fashion image database. *Proceedings of International Joint Conference on Neural Networks (IJCNN'98)*, pages 237–240, 1998.

[5] N. Nasarabadi and R.A. King. Image coding using vector quantization: A review. *IEEE Trans. Commun.*, 36(8):957–971, 1988.

[6] L. Xu. Rival penalized competetive learning, finite mixture, and multisets clustering. *Proc. International Joint conference on Neural Networks*, II:2525–2530, May 5-9 1998.

[7] L. Xu, A. Krzyzak, and E. Oja. Rival penalized competetive learning for clustering analysis, rbf net, and curve detection. *IEEE Trans. on Neural Networks*, 4(4):636–649, July 1993.

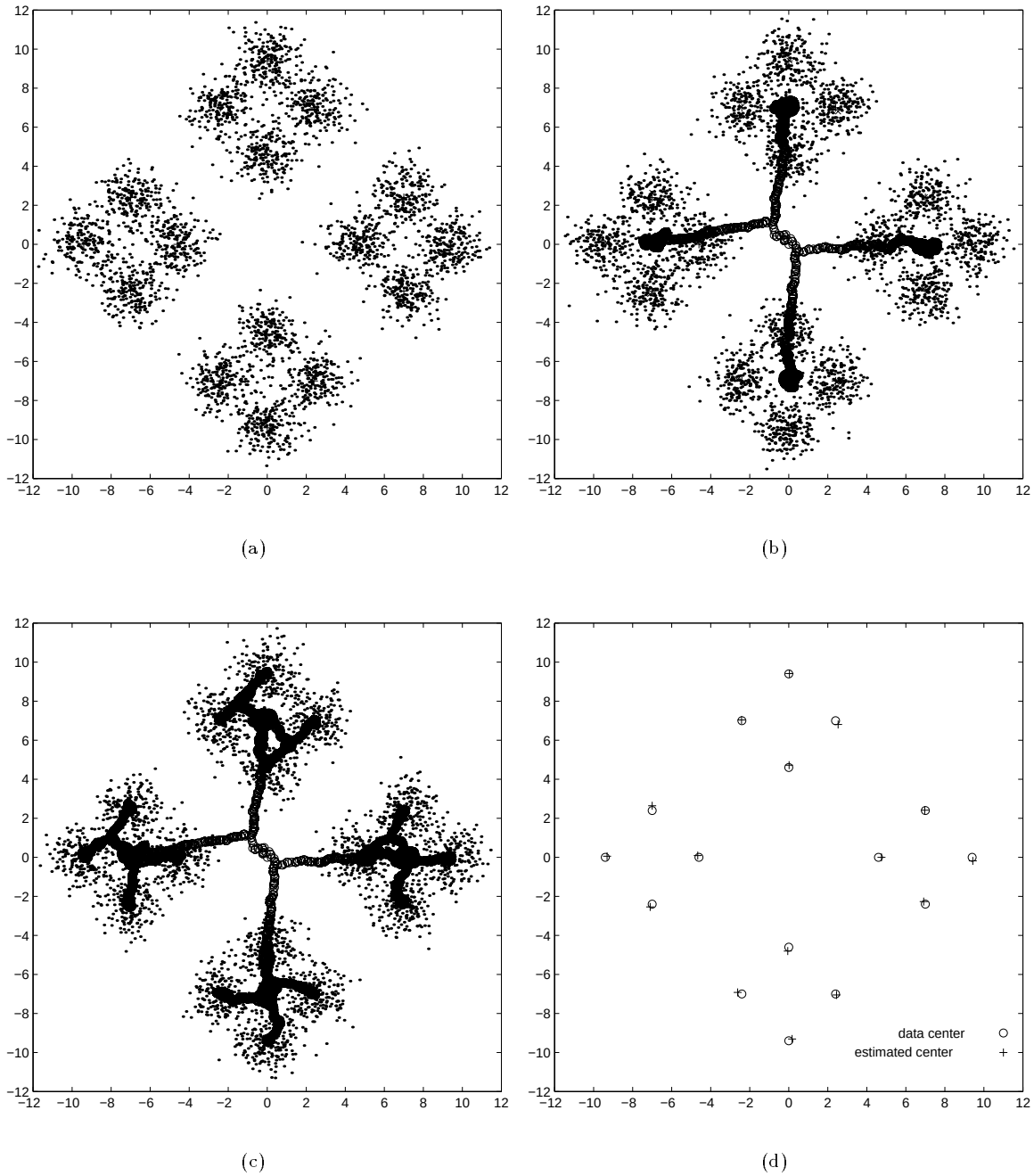



Figure 5: 2-level multiresoution clustering. (a) original data set, (b) one-level BCL clustering, (c) two-level BCL clustering, and (d) is the final cluster center (in circle) along with the estimated centers (in cross) returned by the BCL algorithms.