Egyptian Informatics Journal (2013) 14, 183–194

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

Adaptive competitive learning neural networks

Ahmed R. Abas [*]

Department of Computer Science, College of Computer in Lith, Umm Al-Qura University, Makka Al-Mukarrama, Saudi Arabia
Department of Computer Science, Faculty of Computers and Informatics, Zagazig University, Zagazig, Egypt

Received 30 April 2013; accepted 26 August 2013
Available online 14 September 2013

KEYWORDS

Adaptive competitive learning neural networks;
Conscience learning;
Dead neurons;
Number of clusters;
Robust clustering

Abstract In this paper, the adaptive competitive learning (**ACL**) neural network algorithm is proposed. This neural network not only groups similar input feature vectors together but also determines the appropriate number of groups of these vectors. This algorithm uses a new proposed criterion referred to as the *ACL* criterion. This criterion evaluates different clustering structures produced by the **ACL** neural network for an input data set. Then, it selects the best clustering structure and the corresponding network architecture for this data set. The selected structure is composed of the minimum number of clusters that are compact and balanced in their sizes. The selected network architecture is efficient, in terms of its complexity, as it contains the minimum number of neurons. Synaptic weight vectors of these neurons represent well-separated, compact and balanced clusters in the input data set. The performance of the **ACL** algorithm is evaluated and compared with the performance of a recently proposed algorithm in the literature in clustering an input data set and determining its number of clusters. Results show that the **ACL** algorithm is more accurate and robust in both determining the number of clusters and allocating input feature vectors into these clusters than the other algorithm especially with data sets that are sparsely distributed.

© 2013 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo University.

1. Introduction

Cluster analysis is an important task in pattern recognition. It is interested in grouping similar feature vectors in an input data set into a number of clusters. Feature vectors in one cluster are similar to each other more than other feature vectors in the other clusters. Different clustering algorithms are proposed in the literature such as the competitive neural networks (**CNN**) and the finite mixture models (**FMM**) [1–5]. The neurons of the competitive networks learn to recognize groups of similar input feature vectors. The **FMM** produces a certainty estimate of the membership of each feature vector to each one of the clusters in the input data set. However, these algorithms have some limitations. First, they produce suboptimal results because they converge to the nearest local optima of the objective function to the starting point [6]. Second, they produce biased values for cluster centers when clusters are poorly separated or when cluster sizes are not balanced, i.e.,

* Address: Department of Computer Science, Faculty of Computers and Informatics, Zagazig University, Zagazig, Egypt. Tel.: +966 580016325, +20 1009013037.
E-mail addresses: arabas@zu.edu.eg, armohamed@uqu.edu.sa.

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

data are sparsely distributed [7,8]. These algorithms require the number of clusters to be given [5].

Determining the optimal number of clusters for an input data set is a difficult problem in cluster analysis [5,9]. Several criteria are proposed in the literature for determining the number of clusters with **FMM** [see for review, [10–12]]. However, these criteria tend to overestimate the number of clusters when cluster shapes are not Gaussian [8] and underestimate the number of clusters when clusters are overlapping or when the size of the given data set is small [13]. Also, these criteria have poor performance with sparsely distributed data [10,11].

On the other hand, there is no criterion proposed in the literature for automatic determination of the number of neurons, which represent clusters, in the **CNN** to the best of our knowledge. Alternatively, the number of neurons in the **CNN** is determined manually by removing the neurons that have not win the competition in representing the input feature vectors after convergence [14]. These neurons are called the dead neurons [5]. However, this method is sensitive to the selection of the learning rate [5]. Another method determines the number of neurons or clusters by finding the difference between the synaptic weight vectors of the **CNN,** and if the difference between the weight vectors is greater than a specified value, then the number of clusters is increased by one, otherwise the clusters are merged [15].

In this paper, a new algorithm that is referred to as the adaptive competitive learning neural network (**ACL**) algorithm is proposed to integrate the unsupervised learning and the optimization of the **CNN**. It learns and evaluates different **CNN** architectures for clustering an input data set using a new proposed criterion that is referred to as the *ACL* criterion. The **CNN** architecture corresponds to the minimum value of the *ACL* criterion is considered the most efficient **CNN** architecture, in terms of its complexity. This **CNN** is composed of the minimum number of neurons that represent compact and balanced clusters in the input data set. These clusters have the minimum within-component variation among them. The rest of this paper is organized as follows: Section 2 presents the architecture of the **CNN**. Section 3 presents the proposed criterion and the proposed **ACL** algorithm. Section 4 presents an evaluation and a comparison study of the **ACL** algorithm and the **EMCE** algorithm [12]. This study shows the performance of each algorithm in determining the optimal number of clusters and allocating input feature vectors into these clusters using data sets of different properties. Section 5 presents discussion of the results obtained. Finally, Section 6 presents the conclusions and the future work.

2. The architecture of the CNN

The **CNN** is a single-layer neural network in which each output neuron is fully connected to the input nodes. The output neurons compete to become active when an input feature vector is presented to the network [1,2,5,6,16]. The activation of the neuron with the largest net input is set to 1 and the activation of all other neurons is set to 0. This condition is known as "winner takes all." If a neuron wins the competition, its weight vector is updated according to the competitive learning rule shown in the equation

$$w_j(n+1) = w_j(n) + \xi(n)(p_i - w_j) \tag{1}$$

where ξ is the learning rate and w_j is the weight vector of the winning neuron j for the input feature vector p_i. The network initializes the weight vectors of its neurons randomly. The neurons learn by moving their weights toward the input feature vectors when presented to the network. Feature vectors in the data set are presented randomly to the network a number of times called epochs. After training, each output neuron should represent a cluster in the input data set by moving its own synaptic weight vector to the center of that cluster. In other words, the neurons in a competitive layer distribute themselves in the feature space to recognize frequently presented input feature vectors to the network. Therefore, the **CNN** performs clustering for the input data set. However, the performance of the **CNN** is dependent on the number of the output neurons and the initialization of their weight vectors [6]. Different initial weight vectors may lead to different final clusters because the update rule in Eq. (1) only moves the weight vector of the winning neuron toward its nearest feature vectors. Another drawback of the **CNN** is that it may split one cluster into many small clusters [6]. The architecture for the **CNN** is shown in Fig. 1.

The input vector p and the input weight matrix W are fed to block $||dist||$. The number of features of the input vector is R, while the number of neurons of the **CNN** is S. The output of block $||dist||$ is a vector having S elements. The elements are the negative of the Euclidean distances between the input vector p and vectors formed from the rows of the input weight matrix W. The net input of a competitive layer n is composed by finding the negative distance between input vector p and the weight vectors and adding the biases b. When all biases are zero, the maximum net input for any neuron is 0. This occurs when the input vector p equals that neuron's weight vector. The competitive transfer function block CTF has the net input vector as input for the layer and produces neuron outputs of 0 for all neurons except for the winner whose output is 1. The winner neuron is the one with the most positive element of the net input vector. If all the biases are zero, then the neuron whose weight vector is the nearest to the input vector, i.e. the neuron with the least negative net input, wins the competition and its output becomes 1.

During learning of the weight vectors in the **CNN,** a problem referred to as the dead neurons occurs when some neurons may not get allocated or win the competition for almost all feature vectors in the input data set. This happens when the weight vectors of some neurons start out far from any input feature vectors, and therefore, these neurons never win the

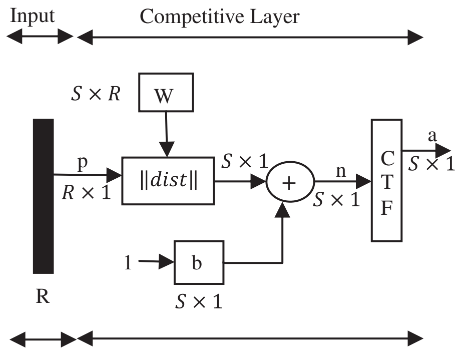

Figure 1 The architecture of the Competitive neural network.

Table 1 A comparison of the ACL and the EMCE algorithms in clustering an input data set and determining its number of clusters. Shaded cells contain the correct values for the number of clusters and the highest values for the NMI measure for each data set.

Data	ACL				EMCE			
	k		NMI		k		NMI	
	Opt.	Avg. ± Std	Opt.	Avg. ± Std.	Opt.	Avg. ± Std.	Opt.	Avg. ± Std.
First data	5	4.84 ± 0.37	1	0.93 ± 0.07	5	9.95 ± 6.99	1	0.8 ± 0.11
Second data	3	3 ± 0	1	0.98 ± 0.04	3	12.98 ± 6.53	1	0.74 ± 0.15
Iris	3	2.97 ± 0.17	0.76	0.72 ± 0.05	3	2.98 ± 0.14	0.9	0.89 ± 0.03
Seeds	3	3 ± 0	0.71	0.7 ± 0.02	3	2.98 ± 0.14	0.85	0.81 ± 0.06
Wine	3	3 ± 0.14	0.88	0.75 ± 0.06	3	6.97 ± 6.68	0.97	0.62 ± 0.13
Breast Cancer	2	2.47 ± 0.85	0.79	0.78 ± 0.11	4	3.72 ± 1.9	0.42	0.48 ± 0.09
Pima Indians Diabetes	2	1.98 ± 0.14	0.06	0.07 ± 0.02	1	1 ± 0	0	0 ± 0
Glass Identification	2	2.06 ± 0.24	0.76	0.86 ± 0.13	1	1 ± 0	0	0 ± 0

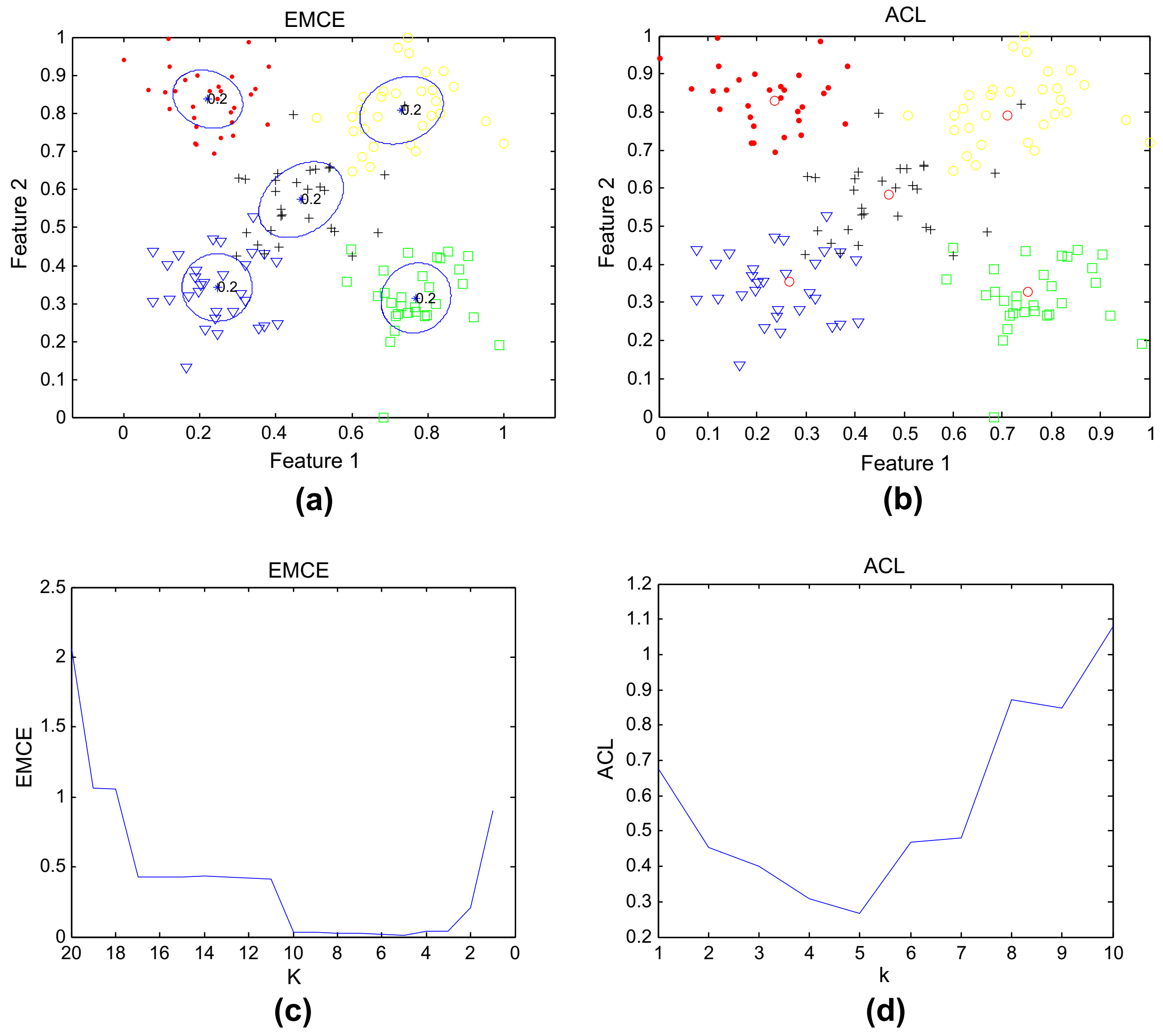

Figure 2 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the First data set.

competition, and their weight vectors do not get to learn. These neurons do not perform a useful function in the **CNN**. In order to prevent the occurrence of the dead neurons, biases are used to give neurons that rarely win the competition an advantage over neurons that win often. A positive bias added to the negative distance makes a distant neuron more likely to win. To do this job, the percentage of times the output of each neuron is 1 is stored. This percentage is used to update the biases, so that the biases of frequently active neurons will be smaller, and the biases of infrequently active neurons will be larger. The learning rate for updating the biases is called the conscience learning rate, and its value is set smaller than the competitive learning rate ξ. In the experiments shown in this paper, the competitive learning rate is set to 0.01, and the conscience learning rate is set to 0.001, respectively. Integrating biases learning with the competitive learning for the neurons of the **CNN** is called learning with conscience [17]. This learning process allows the infrequently active neuron to respond and move toward more input feature vectors. Eventually, the neuron will respond to an equal number of feature vectors as other neurons. As a result of learning with conscience, the dead neurons problem is resolved, and each neuron in the **CNN** is forced to represent roughly the same number of input feature vectors [18].

3. The proposed adaptive competitive learning neural network

The adaptive competitive learning (**ACL**) neural network algorithm uses a new proposed *ACL* criterion to determine the optimal number of output neurons in the **CNN** for clustering an input data set. This criterion is based on the theory that the best **CNN** architecture for clustering an input data set should produce a cluster structure that is composed of dense, well-separated and balanced clusters that have the minimum number of parameters to be estimated. To realize this theory, the *ACL* criterion selects the **CNN** that represents minimum within-cluster variations and minimum value for the product of the relative weights of clusters in the given data set. To introduce the notation, let $\mathbf{D} = \{p_1, p_2, \ldots, p_n\}$ be a given data set that consists of n feature vectors that are independently and identically distributed in R-feature space. The values on each feature are scaled such that they range from 0 to 1. This re-

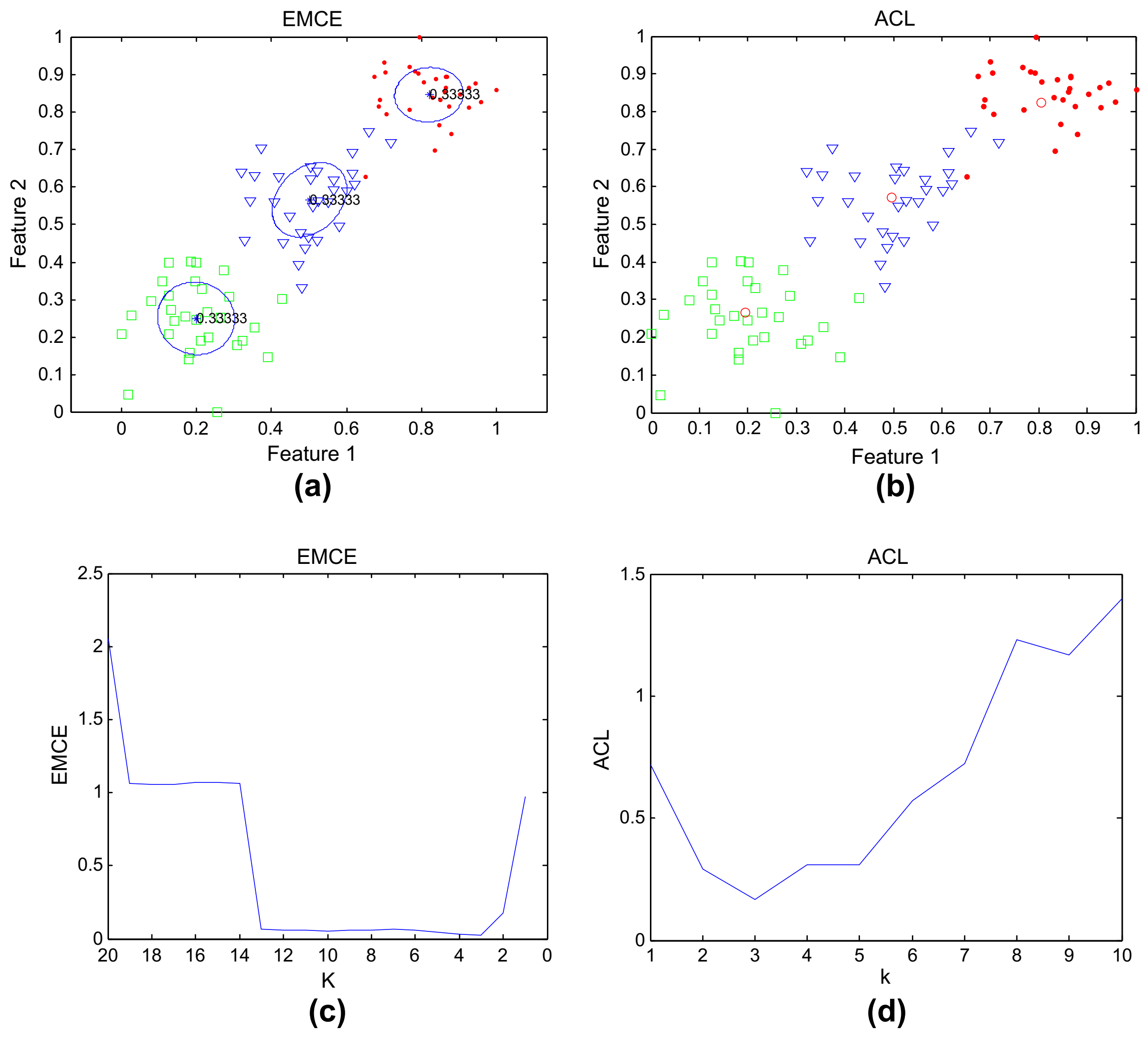

Figure 3 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the Second data set.

duces the sparsity of the input data and increases the accuracy of estimating its cluster structure [10]. The cluster structure is revealed using the **CNN** that represents the input data set. Each neuron in output competitive layer is assumed to represent a cluster in the input data set. Then, using a **CNN** that contains k neurons in its output layer, the average within-cluster variations in this data set are defined as:

$$E(k) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}\|p_i - w_j\|^2 \tag{2}$$

where $p_i \in \mathbf{D}$, w_j is the weight vector of neuron j and r_{ij} is set to 1 if neuron j wins the competition when input vector p_i is presented to the network, otherwise it is set to 0. The relative weight of a certain cluster in the data set represented by a neuron j in the **CNN** is defined as:

$$\pi_j = \frac{1}{n}\sum_{i=1}^{n} r_{ij} \tag{3}$$

Then, the proposed ACL criterion for evaluating different architectures of the **CNN** that are composed of different numbers of neurons is defined as:

$$ACL(k) = E(k) + \frac{1}{2k_{\max}}\sum_{j=1}^{k}\log_{10}\left(\frac{1}{\pi_j}\right) \tag{4}$$

where $k_{\max}$ is the maximum number of neurons in the output layer in the **CNN**. The optimal architecture for the **CNN** that is composed of the optimal number of output neurons for best clustering a given data set corresponds to the minimum value of the ACL criterion. Minimizing the first term in the ACL criterion produces the minimum within-cluster variation, which in turn results in the most compact and dense clusters. In addition, minimizing the second term produces the minimum product of cluster relative weights that corresponds to the minimum number of well-separated clusters that have roughly equal weights. This in turn results in the most efficient **CNN** in terms of its complexity. Therefore, minimizing the ACL criterion produces the most efficient **CNN** that represents compact, well-separated and balanced clusters in the given data set. Starting with a **CNN** with $k_{\max}$ neurons the first term of the ACL criterion will be too small, while the second term has maximum value. As k decreases, the first term increases because the sizes of the clusters increase, while the second term decreases because relative weights of the clusters become

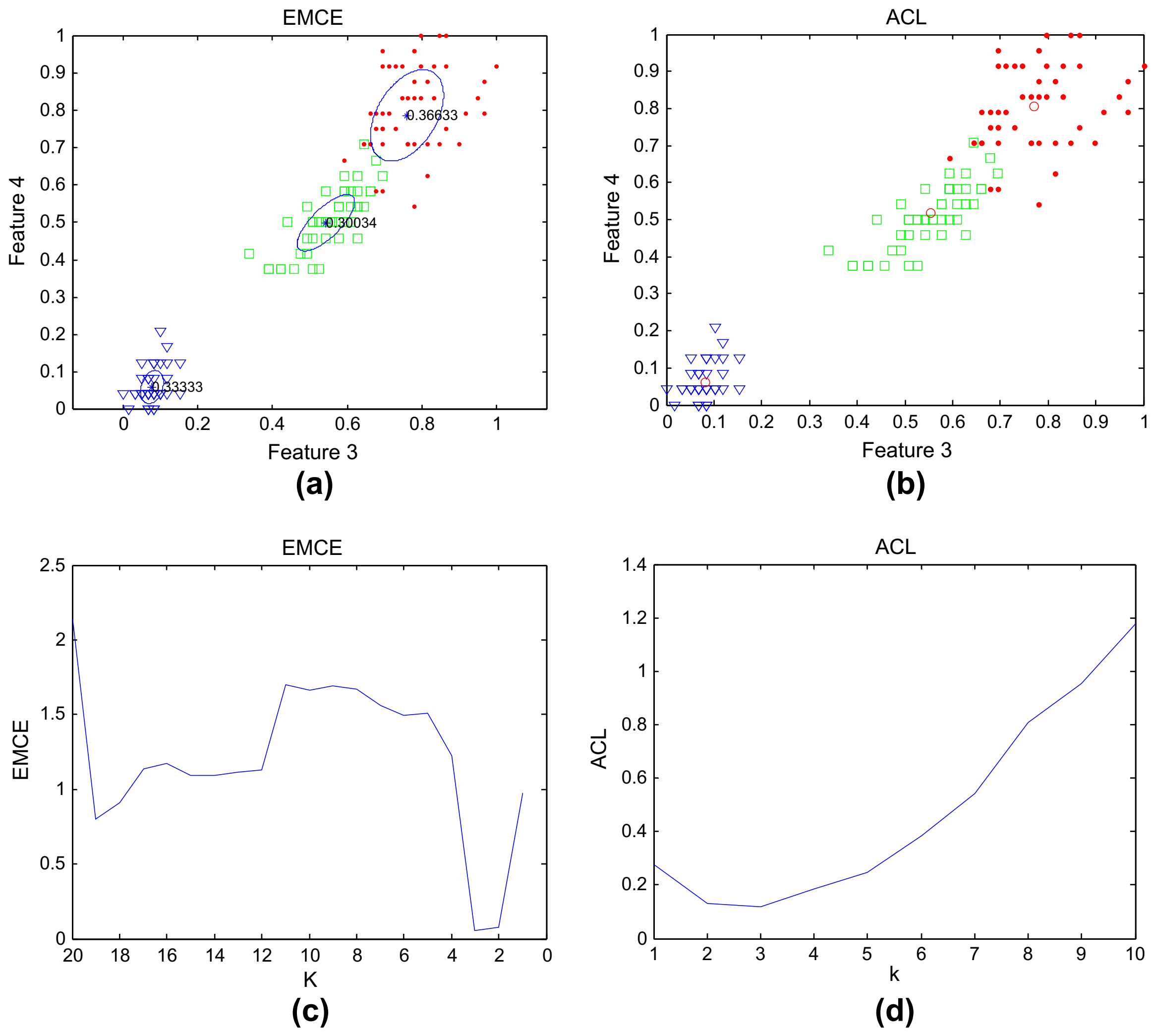

Figure 4 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the Iris data set.

larger. The minimum *ACL* criterion value assures the best compromise between compactness of the clusters and their number, i.e., complexity of the **CNN**.

The **ACL** algorithm uses learning with conscience [17] in order to resolve the dead neurons problem during competitive learning of the weight vectors of the output neurons. In addition, this learning process assures that these neurons represent balanced clusters, i.e., clusters of roughly equal number of input feature vectors. The algorithm starts with a **CNN** of large number of neurons $k_{\max}$ that is set to 10 in the experiments shown in this paper. Input feature vectors are presented randomly to the network for a number of epochs. The number of epochs in experiments shown in this paper is set to 10. Learning with conscience [17] is used to learn weight vectors of the output neurons. After convergence, the *ACL* criterion value of the current **CNN** architecture is computed. The number of neurons is decremented, and weight vectors of the new **CNN** architecture are learned. This process continues until there is one neuron in the **CNN** architecture. Finally, the **CNN** architecture that has the minimum *ACL* criterion value is considered the optimal **CNN** for clustering the input data set, and its number of neurons is considered the optimal number of clusters in this data set. This **CNN** has the minimum number of neurons that represent well-separated and balanced clusters. These clusters have the minimum within-cluster variation. This **CNN** is efficient because it has a small number of neurons, i.e., small complexity and it represents compact, balanced and well-separated clusters. Finally, the steps of the proposed **ACL** algorithm are shown as follows:

```
Program CNN = ACL(data)
Step 0. Normalize the values of each input data feature to range from 0 to 1.
Step 1. Within-cluster variations, product of the relative weights of clusters and the number of neurons should be minimized during optimization.
Step 2. Start with a CNN that has a large number of neurons $k = k_{\max}$ and empty BestCNN.
Step 3. do{
    Use learning with conscience to learn weight vectors of the neurons in the current CNN.
Step 4. Present the input feature vectors to the trained CNN and obtain the corresponding output vectors and cluster indexes.
Step 5. Compute the average within-cluster variations $E(k)$ and the relative weights of clusters $\pi_j$, $j = 1{:}k$.
Step 6. Compute the $ACL(k)$ criterion value (see, Equation 4) for the current CNN.
Step 7. If (BestCNN is Empty or $ACL(k) < $ BestCNN.$ACL$) then save the current weight vectors and $ACL(k)$ as BestCNN.
    else create a new CNN with $k = k - 1$
    end If
} while($k \geqslant 1$)
Step 8. Use the BestCNN as the optimal CNN for clustering the input data set.
Step 9. Stop.
```

4. Experimental results and evaluation

The performance of the **ACL** algorithm is compared to the performance of the recently proposed **EMCE** algorithm [12] in clustering an input data set and determining the number of clusters in it. The **EMCE** algorithm proves superiority to other algorithms in the literature in clustering an input data set and determining the number of clusters in it [12]. It uses mixture models and a criterion that is based on likelihood and mutual information theory for evaluating different mixture models with different numbers of clusters. The compared algorithms are implemented, and experiments are carried out using the MATLAB software. Different data sets with different cluster structures are used. These data sets are described in Section 4.1. The measure used to quantify how good the clustering results obtained from each algorithm is described in Section 4.2.

4.1. Data sets

Data sets used in the experiments shown in this paper have different cluster structures in terms of number of clusters, cluster separation and data sparsity in the feature space. These data sets are described as follows:

4.1.1. The first data set

This data set is artificially generated such that it consists of 150 feature vectors each of which is a vector in 10-feature space. These feature vectors are generated from five separated Gaussian-shape clusters with equal probabilities. The centers of these clusters are $\boldsymbol{\mu}_1 = [2,2,2,2,2,2,2,2,2,2]^T$, $\boldsymbol{\mu}_2 = [6,2,2,2,6,6,2,2,6,6]^T$, $\boldsymbol{\mu}_3 = [2,6,6,6,2,2,6,6,2,2]^T$, $\boldsymbol{\mu}_4 = [4,4,4,4,4,4,4,4,4,4]^T$ and $\boldsymbol{\mu}_5 = [6,6,6,6,6,6,6,6,6,6]^T$, while their covariance matrices are identical and equal to $\boldsymbol{\Sigma} = 0.5\mathbf{I}_{10}$. The purpose of using this data set is to test the algorithms compared when data clusters are separated and when the number of features is large compared to the number of feature vectors, i.e., the data set is sparsely distributed.

4.1.2. The second data set

This data set is artificially generated such that it consists of 90 feature vectors each of which is a vector in 10-feature space. These feature vectors are generated from three poorly separated Gaussian-shape clusters with equal probabilities. The centers of these clusters are $\boldsymbol{\mu}_1 = [0,0,0,0,0,0,0,0,0,0]^T$, $\boldsymbol{\mu}_2 = [-2,-2,-2,-2,-2,-2,-2,-2,-2,-2]^T$ and $\boldsymbol{\mu}_3 = [2,2,2,2,2,2,2,2,2,2]^T$, while their covariance matrices are identical and equal to $\boldsymbol{\Sigma} = \mathbf{I}_{10}$. The purpose of using this data set is to test the algorithms compared when data clusters are poorly separated and when the number of features is large compared to the number of feature vectors i.e., the data set is sparsely distributed. Both of the first and the second data sets are used in the literature in comparing different clustering algorithms [12].

4.1.3. The Iris data set

It consists of 150 feature vectors each of which is a vector in four-feature space. These feature vectors represent three clusters of equal sizes. Two clusters are overlapped in the data space. The purpose of using this data set is to test the algorithms compared when data clusters are poorly separated and when the number of features is small.

4.1.4. The Seeds data set

It consists of 210 feature vectors each of which is a vector in seven-feature space. These feature vectors represent three clusters of equal sizes. These clusters are partially overlapped in the data space. The purpose of using this data set is to test the algorithms compared when data clusters are poorly separated and when the number of features is moderate.

4.1.5. The Wine data set

It consists of 178 feature vectors each of which is a vector in 13-feature space. These feature vectors represent three clusters whose sizes are 59, 71 and 48 feature vectors. The clusters are separated in the data space. The purpose of using this data set is to test the algorithms compared when data clusters are separated and when the number of features is large compared to the number of feature vectors.

4.1.6. The Breast Cancer data set

It consists of 683 feature vectors each of which is a vector in 9-feature space. These feature vectors represent two unbalanced clusters whose sizes are 444 and 239 feature vectors, respectively. To be composed of balanced clusters, the size of this data set is reduced to 478 feature vectors that represent two clusters of equal sizes. The clusters are overlapping in the data space. The purpose of using this data set is to test the algorithms compared when data clusters are overlapping and when the number of features is moderate compared to the number of feature vectors.

4.1.7. The Pima Indians Diabetes data set

It contains 768 feature vectors each of which is a vector in eight-feature space. These feature vectors represent two unbalanced clusters whose sizes are 500 and 268 feature vectors, respectively. To be composed of balanced clusters, the size of this data set is reduced to 536 feature vectors that represent two clusters of equal sizes. These clusters are largely overlapping. Correlations between different pairs of features of this data set are too weak. The purpose of using this data set is to test the algorithms compared when data clusters are largely overlapping and when the number of features is moderate compared to the number of feature vectors.

4.1.8. The Glass Identification data set

It contains 214 feature vectors each of which is a vector in nine-feature space. These feature vectors represent two unbalanced clusters whose sizes are 163 and 51 feature vectors, respectively. To be composed of balanced clusters, the size of this data set is reduced to 102 feature vectors that represent

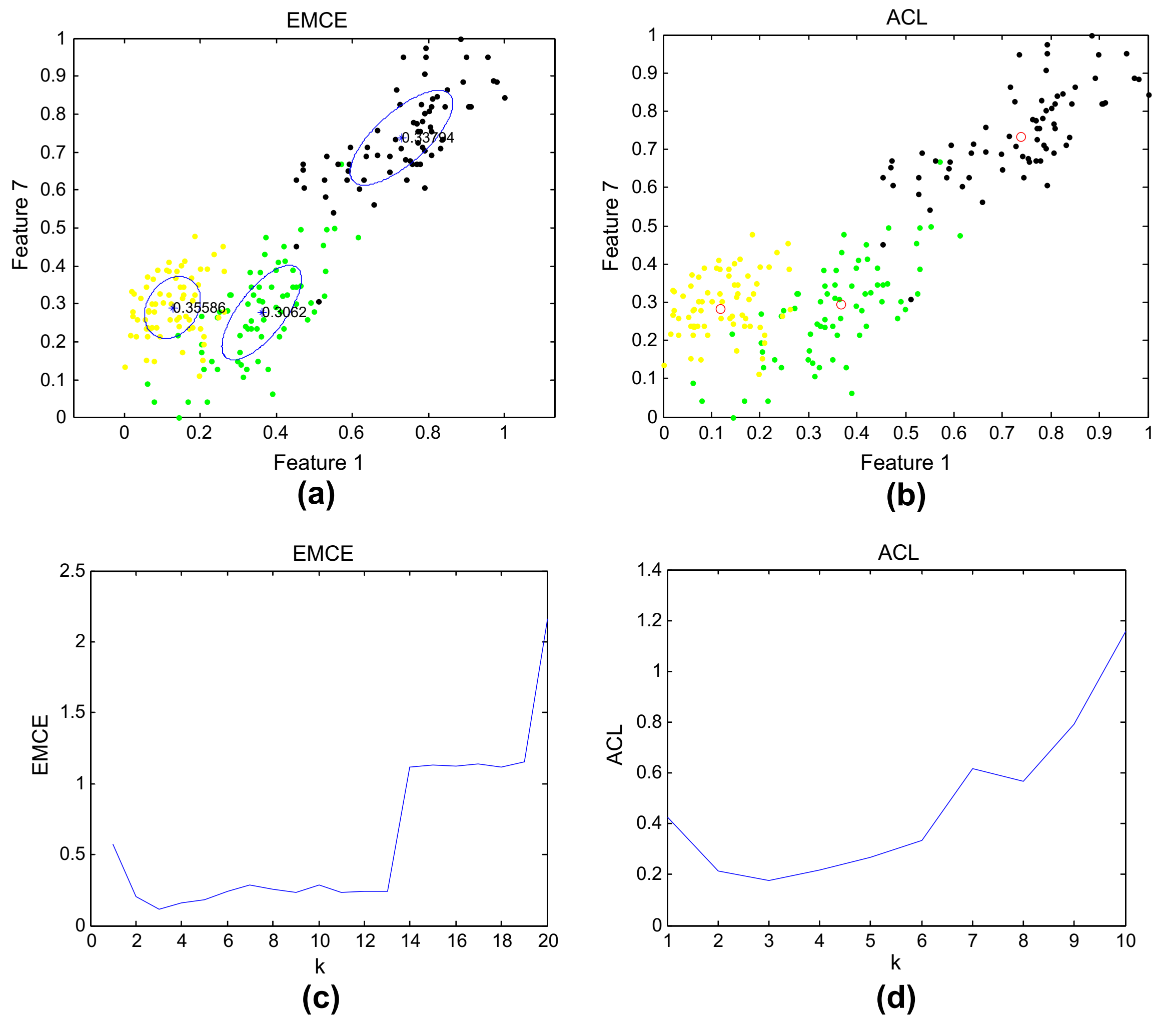

Figure 5 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the Seeds data set.

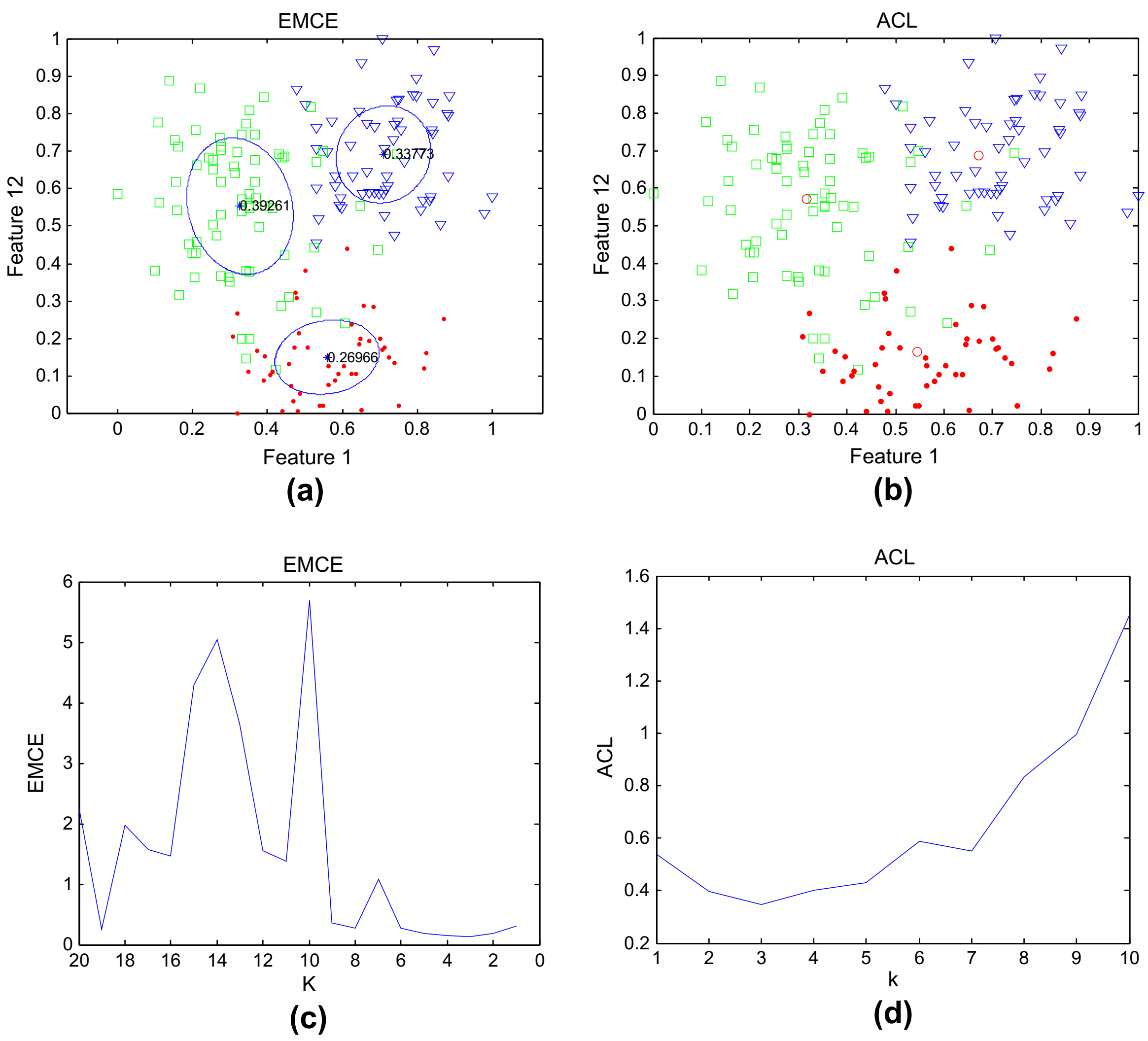

Figure 6 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the Wine data set.

two clusters of equal sizes. These clusters are separated in the feature space. The purpose of using this data set is to test the algorithms compared when data clusters are separated and when the number of features is large compared to the number of feature vectors i.e., the data are sparsely distributed. All these real data sets are commonly used in classification analysis [19].

4.2. The evaluation criterion for clustering results

The Mutual Information is a symmetric measure to quantify the statistical information shared between two distributions [20]. Therefore, this measure is used to quantify how good the clustering results obtained for a certain data set is by comparing it to the true classification of this data set [21]. Let **X** and **Y** be two random variables represent the true class labels [1..m] for a certain data set and the cluster labels [1..k] resulting from an algorithm for the same data set, respectively. The mutual information between **X** and **Y** is defined as $I(\mathbf{X};\mathbf{Y}) = \sum_{i=1}^{m}\sum_{j=1}^{k} P_{ij}\log_2(P_{ij}/P_iP_j)$, where P_{ij} is the probability that a member of cluster j belongs to class i, P_i is the probability of class i and P_j is the probability of cluster j. Since this measure is not bounded by the same constant for all data sets, a normalized version that ranges from 0 to 1 is proposed for easier interpretation and comparison [21]. This normalized version is called the Normalized Mutual Information (**NMI**) and is computed as follows:

$$NMI(\mathbf{X};\mathbf{Y}) = \frac{I(\mathbf{X};\mathbf{Y})}{\sqrt{H(\mathbf{X})H(\mathbf{Y})}} \tag{5}$$

where $H(\mathbf{X})$ and $H(\mathbf{Y})$ denote the entropy of **X** and **Y**. The **NMI** has the value of 1 when there is a one to one mapping between the clusters obtained and the true classes (i.e., $k = m$) of a given data set. Since this measure is not biased toward large k, it is preferred to compare different data partitions [21,22].

5. Discussion of results

Table 1 shows the performances of the algorithms compared with each one of the data sets used. The performance of each algorithm is evaluated by the values of the **NMI** criterion, and

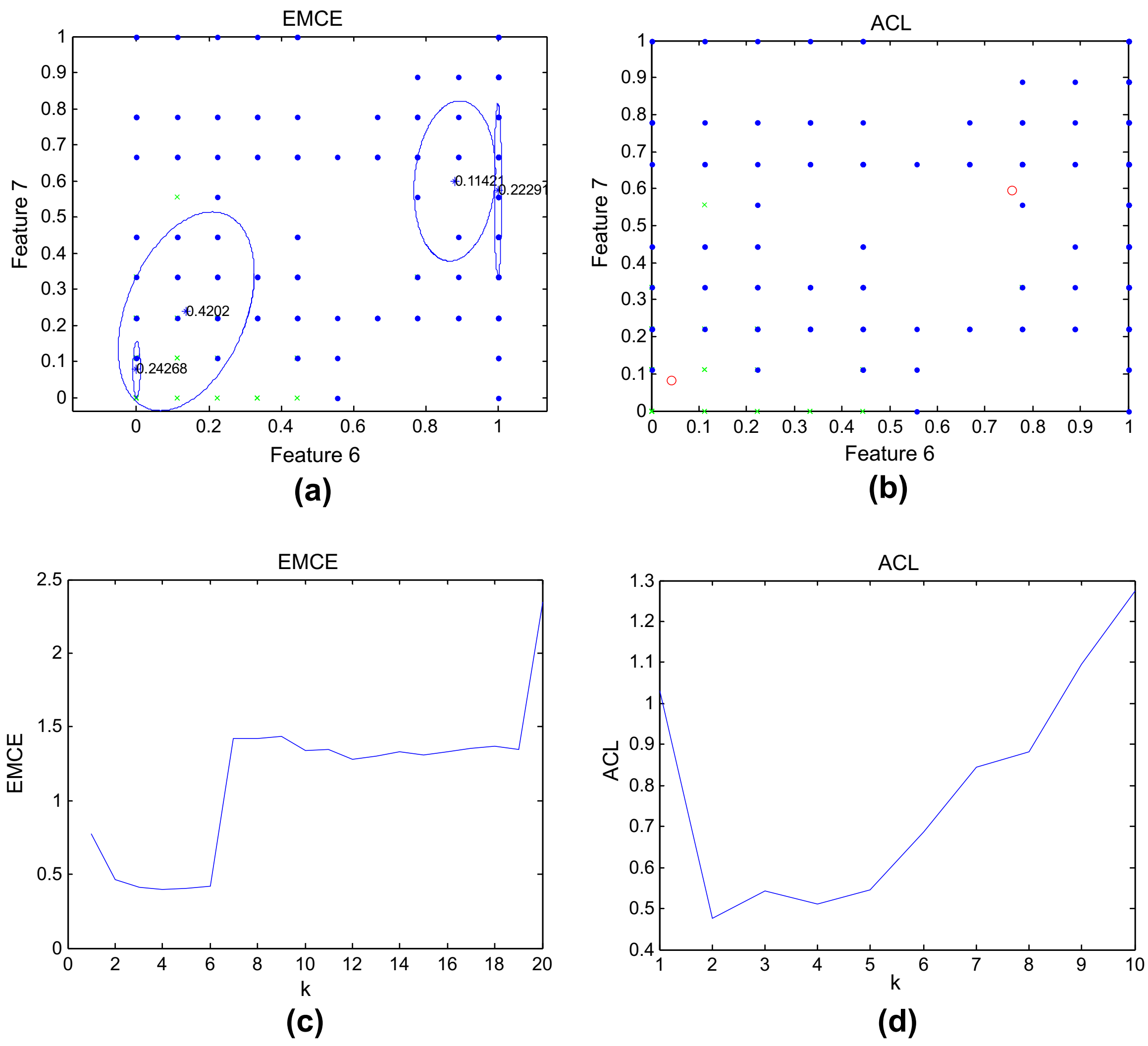

Figure 7 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the Breast Cancer data set.

the number of clusters corresponding to the optimal value of the criterion used by the algorithm and the average ± the standard deviation resulting from 100 experiments. Each experiment has different random initialization values of the neuron weight vectors in the **ACL** algorithm and the mixture component parameters in the **EMCE** algorithm. This repetition of the experiments removes the effect of initialization values on the results of the algorithms [11,12]. The shaded cells in this table represent the maximum values of the **NMI** among all algorithms and the correct number of clusters with each data set. Figs. 2–9(a and b) show examples of the **FMM**s obtained from the **EMCE** algorithm, represented by blue ellipses, and neuron weight vectors (cluster centers) obtained from the **ACL** algorithm, represented by red circles, with each one of the eight data sets used. The ellipses in these figures are isodensity curves of each component in the **FMM**. These figures (c and d) also show the distribution of the **EMCE** and the **ACL** criteria against the number of clusters k through the runtime of the corresponding algorithms.

Table 1 shows that the **ACL** algorithm results in the correct optimal number of clusters in all data sets used, while the **EMCE** algorithm fails in producing the correct optimal number of clusters with the Breast Cancer, the Pima Indians Diabetes and the Glass Identification data sets. In addition, the **ACL** algorithm produces robust results as its average number of clusters over 100 experiments is approximately equal to the correct number of clusters, while the standard deviation is too small with all data sets used. On the other hand, the **EMCE** algorithm does not produce robust results as its average number of clusters over 100 experiments is far from the correct number of clusters, and its standard deviation is large with all data sets used except the Iris and the Seeds data sets.

Regarding the clustering results, the **ACL** algorithm produces the highest and the most robust results of the **NMI** measure with all data sets except the Iris and the Seeds data sets. With the Iris and the Seeds data sets, the **EMCE** algorithm produces the highest and the most robust results for the **NMI** measure as these data sets have overlapping among their clusters. In general, results of the **ACL** algorithm are accurate and robust in determining the number of clusters in all data sets used. With respect to clustering results, the **ACL** algorithm is accurate and robust with data sets that do not

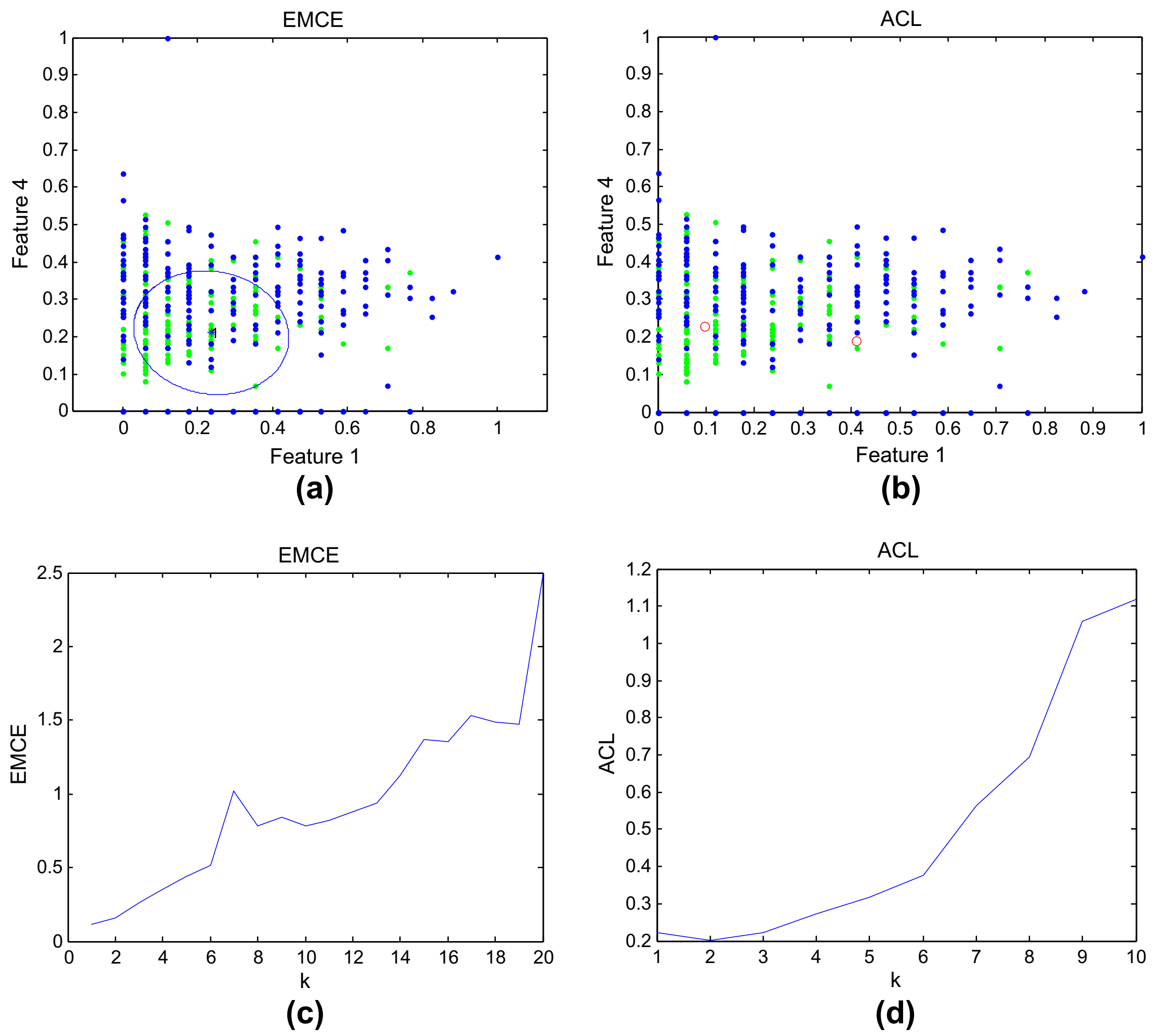

Figure 8 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the Pima Indians Diabetes data set.

have overlapping among their clusters and are sparsely distributed. These results show that the **ACL** algorithm is less sensitive to the curse of dimensionality than the **EMCE** algorithm. This is because the **ACL** algorithm depends on the Euclidean distance in the feature space, while the **EMCE** algorithm depends on the probability density function in clustering an input feature vector.

Figs. 2–9 show that the **ACL** algorithm produces correct and robust clustering results either the number of clusters or the allocation of input feature vectors to these clusters with all data sets used. In addition, drawing the *ACL* criterion against *k* shows that it is convex and does not have overshooting or flat areas as the *EMCE* criterion does. This shows that the *ACL* criterion is not only accurate but also robust especially with sparsely distributed data sets.

6. Conclusions and future work

In this paper, the competitive neural network (**CNN**) is developed in order to be able to cluster an input data set and determine its number of clusters. The resulting algorithm is referred to as the adaptive competitive learning (**ACL**) neural network. It is based on a new proposed criterion for evaluating the clustering structure produced by the **CNN**, called the *ACL* criterion. This criterion evaluates the clustering structure by compromising the compactness of the clusters produced by the relative weights of these clusters. It prefers the cluster structure that is composed of the minimum number of clusters that are compact, well-separated and balanced in their sizes. The **ACL** algorithm is compared with the recently proposed **EMCE** algorithm that proves superiority to other algorithm in the literature in clustering an input data set and determining its number of clusters. Different data sets with different cluster structures in terms of number of clusters, cluster separation and data sparsity in the data space are used. Results show that the **ACL** algorithm produces more accurate and robust results than the **EMCE** algorithm especially with data sets that are sparsely distributed. The **ACL** algorithm does not impose a certain cluster shape on the data set, while the **EMCE** imposes the Gaussian shape on data clusters. In addition, it overcomes the problem of the **EMCE** algorithm when the input data are sparsely distributed. This is because the **ACL** algorithm uses

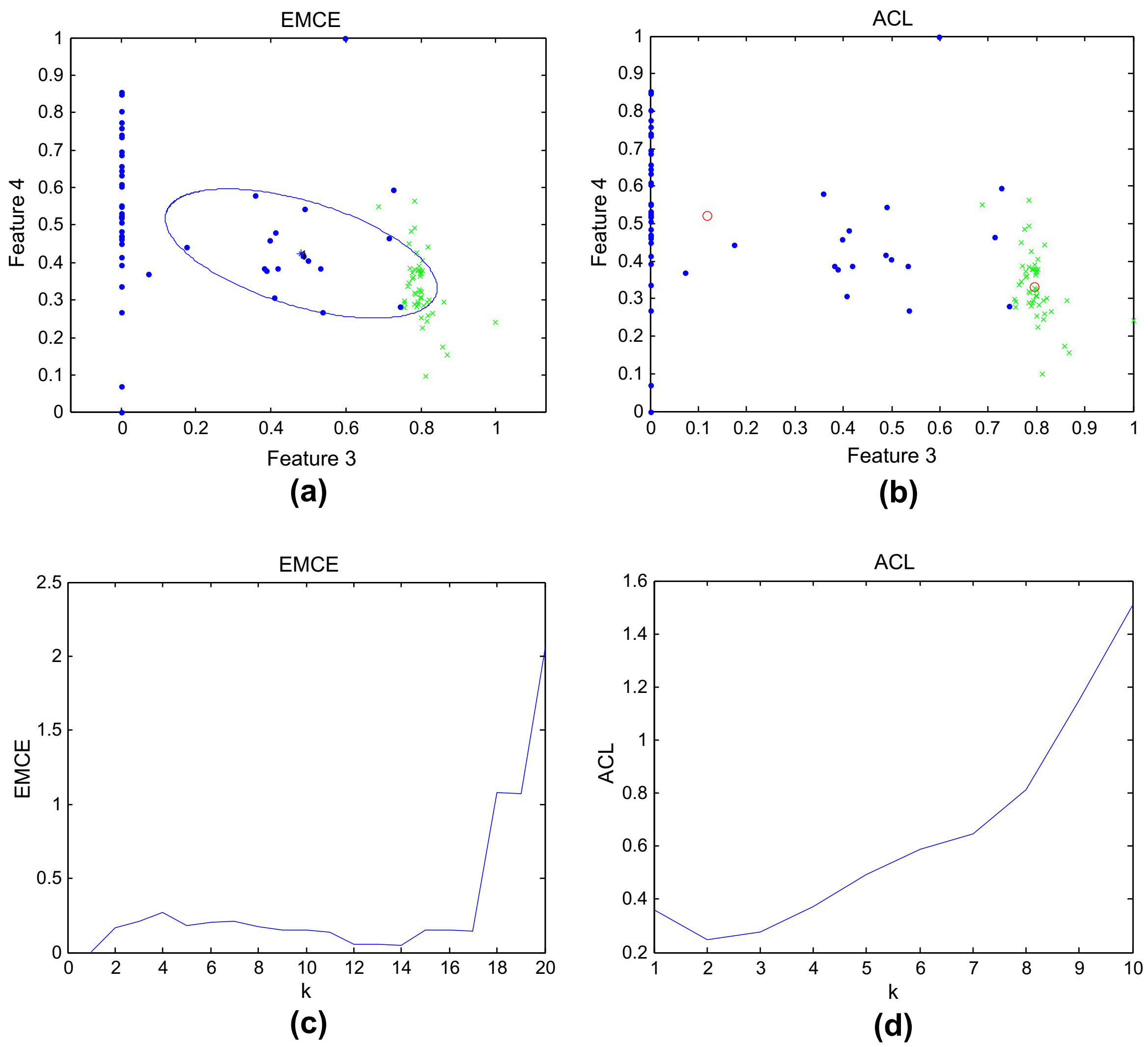

Figure 9 The clusters obtained from the EMCE and the ACL algorithms and the distribution of the criteria used in these algorithms against the number of clusters k with the Glass Identification data set.

the Euclidean distance in the feature space rather than the probability density function as the **EMCE** algorithm does in allocating each input feature vector to a certain cluster. Both of the algorithms compared are similar in producing a single frame for determining the number of clusters and allocating input feature vectors in a certain data set to these clusters.

In the future, the *ACL* criterion may be tested for other competitive learning algorithms such as the fuzzy competitive learning neural networks [23], the fuzzy c-means algorithm [24] and the k-means algorithm [25].

References

[1] Rumelhart DE, Zipser D. Feature discovery by competitive learning. Parallel distributed processing. MIT Press; 1986, p. 151–193.
[2] Kohonen T. Self-organization and associative memory. 2nd ed. Berlin: Springer-Verlag; 1987.
[3] Duda RO, Hart PE, Stork DG. Pattern classification. 2nd ed. USA: John Wiley & Sons; 2001.
[4] Webb A. Statistical pattern recognition. 2nd ed. UK: John Wiley & Sons; 2002.
[5] Budura G, Botoca C, Miclau N. Competitive learning algorithms for data clustering. Electron Energetics J 2006;19(2):261–9.
[6] Lei JZ, Ghorbani AA. Network intrusion detection using an improved competitive learning neural network. In: Proceedings of the second annual conference on communication networks and services research; 2004. p. 190–197.
[7] Chinrungrueng C, Sequin CH. Optimal adaptive K-means algorithm with dynamic adjustment of learning rate. IEEE Trans Neural Networks 1995;6(1):157–69.
[8] Biernacki C, Govaert G. Using the classification likelihood to choose the number of clusters. J Comput Sci Stat 1997; 29(2):451–7.
[9] Guo P, Chen CLP, Lyu MR. Cluster number selection for a small set of samples using the Bayesian Ying-Yang model. J IEEE Trans Neural Networks 2002;13(3):757–63.
[10] Rafat A. An adaptive approach for clustering incomplete data sets: an application to health inequality analysis. Germany: Lambert Academic Publishing; 2010.
[11] Abas AR. An algorithm for unsupervised learning and optimization of finite mixture models. Egypt Inform J 2011;12(1):19–27.

[12] Abas AR. On determining efficient finite mixture models with compact and essential components for clustering data. Egypt Inform J 2013;14(1):79–88.

[13] Yang ZR, Zwolinski M. Mutual information theory for adaptive mixture models. J IEEE Trans Pattern Anal Mach Intell 2001;23(4):396–403.

[14] Xu L, Krzyzak A, Oja AE. Rival penalized competitive learning for clustering analysis. rbf net and curve detection. IEEE Trans Neural Networks 1993;4:636–64.

[15] Sowmya B, Rani BS. Color image segmentation using fuzzy clustering techniques and competitive neural network. J Appl Soft Comput 2011;11:3170–8.

[16] Behnke S, Karayiannis NB. Competitive neural trees for pattern classification. IEEE Trans Neural Networks 1998;9(6):1352–69.

[17] Desieno D. Adding a conscience to competitive learning. In: Proceedings of the second IEEE international conference on, neural networks (ICNN-88), vol. 1; 1988. p. 117–24

[18] Demuth H, Beale M. Self-organizing and learning vector quantization nets. In: Neural network toolbox for use with MATLAB User's Guide, The MathWorks Inc., 2002. <www.mathworks.com>.

[19] UCI Repository of machine learning databases, Irvine, CA: University of California, Department of Information and Computer Science, March 2013. <http://archive.ics.uci.edu/ml/>.

[20] Cover TM, Thomas JA. Elements of information theory. Wiley; 1991.

[21] Strehl A, Ghosh J. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 2002;3:583–617.

[22] Fern XZ, Brodley CE. Random projection for high dimensional data clustering: cluster ensemble approach. In: Proceeding of the 20th international conference on machine learning (ICML 2003); 2003. p. 186–93.

[23] Madiafi M, Bouroumi A. A new fuzzy learning scheme for competitive neural networks. J Appl Math Sci 2012;6(63):3133–44.

[24] Zanaty EA. Determining the number of clusters for kernelized fuzzy C-means algorithms for automatic medical image segmentation. Egypt Inform J 2012;13:39–58.

[25] Ossama O, Mokhtar HMO, El-Sharkawi ME. An extended k-means technique for clustering moving objects. Egypt Inform J 2011;12:45–51.