Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition Letters 25 (2004) 641–654
Pattern Recognition Letters
www.elsevier.com/locate/patrec

A clustering method based on boosting

D. Frossyniotis [a,*], A. Likas [b], A. Stafylopatis [a]

[a] *School of Electrical and Computer Engineering, National Technical University of Athens, 9 Iroon Polytechniou Str., Zographou 15773, Athens, Greece*
[b] *Department of Computer Science, University of Ioannina, 451 10 Ioannina, Greece*

Received 16 July 2003; received in revised form 1 December 2003

Abstract

It is widely recognized that the boosting methodology provides superior results for classification problems. In this paper, we propose the boost-clustering algorithm which constitutes a novel clustering methodology that exploits the general principles of boosting in order to provide a consistent partitioning of a dataset. The boost-clustering algorithm is a multi-clustering method. At each boosting iteration, a new training set is created using weighted random sampling from the original dataset and a simple clustering algorithm (e.g. *k*-means) is applied to provide a new data partitioning. The final clustering solution is produced by aggregating the multiple clustering results through weighted voting. Experiments on both artificial and real-world data sets indicate that boost-clustering provides solutions of improved quality.

Keywords: Ensemble clustering; Unsupervised learning; Partitions schemes

1. Introduction

Unlike classification problems, there are no established approaches that combine multiple clusterings. This problem is more difficult than designing a multi-classifier system: in the classification case it is straightforward whether a basic classifier (weak learner) performs well with respect to a training point, while in the clustering case this task is difficult since there is a lack of knowledge concerning the label of the cluster to which a training point actually belongs. The majority of clustering algorithms are based on the following four most popular clustering approaches: iterative square-error partitional clustering, hierarchical clustering, grid-based clustering and density-based clustering (Halkidi et al., 2001; Jain et al., 2000).

Partitional methods can be further classified into hard clustering methods, whereby each sample is assigned to one and only one cluster, and soft clustering methods, whereby each sample can be associated (in some sense) with several clusters. The most commonly used partitional clustering algorithm is *k*-means, which is based on the square-error criterion. This algorithm is computationally efficient and yields good results if the

* Corresponding author. Tel.: +30-210-7722508; fax: +30-210-7722109.
E-mail addresses: dfros@cslab.ntua.gr (D. Frossyniotis), arly@cs.uoi.gr (A. Likas), andreas@cslab.ntua.gr (A. Stafylopatis).

0167-8655/$ - see front matter
doi:10.1016/j.patrec.2003.12.018

clusters are compact, hyper-spherical in shape and well separated in the feature space. Numerous attempts have been made to improve the performance of the simple *k*-means algorithm by using the Mahalanobis distance to detect hyper-ellipsoidal shaped clusters (Bezdek and Pal, 1992) or by incorporating a fuzzy criterion function resulting in a fuzzy *c*-means algorithm (Bezdek, 1981). A different partitional clustering approach is based on probability density function (pdf) estimation using Gaussian mixtures. The specification of the parameters of the mixture is based on the expectation–minimization algorithm (EM) (Dempster et al., 1977). A recently proposed greedy-EM algorithm (Vlassis and Likas, 2002) is an incremental scheme that has been found to provide better results than the conventional EM algorithm.

Hierarchical clustering methods organize data in a nested sequence of groups which can be displayed in the form of a dendrogram or a tree (Boundaillier and Hebrail, 1998). These methods can be either agglomerative or divisive. An agglomerative hierarchical method places each sample in its own cluster and gradually merges these clusters into larger clusters until all samples are ultimately in a single cluster (the root node). A divisive hierarchical method starts with a single cluster containing all the data and recursively splits parent clusters into daughters.

Grid-based clustering algorithms are mainly proposed for spatial data mining. Their main characteristic is that they quantise the space into a finite number of cells and then they do all operations on the quantised space. On the other hand, density-based clustering algorithms adopt the key idea to group neighbouring objects of a data set into clusters based on density conditions.

However, many of the above clustering methods require additional user-specified parameters, such as the optimal number and shapes of clusters, similarity thresholds and stopping criteria. Moreover, different clustering algorithms and even multiple replications of the same algorithm result in different solutions due to random initializations, so there is no clear indication for the best partition result.

In (Frossyniotis et al., 2002) a multi-clustering fusion method is presented based on combining the results from several runs of a clustering algorithm in order to specify a common partition. Another multi-clustering approach is introduced by Fred (2001), where multiple clusterings (using *k*-means) are exploited to determine a co-association matrix of patterns, which is used to define an appropriate similarity measure that is subsequently used to extract arbitrarily shaped clusters. Model structure selection is sometimes left as a design parameter, while in other cases the selection of the optimal number of clusters is incorporated in the clustering procedure (Smyth, 1996; Fisher, 1987) using either local or global cluster validity criteria (Halkidi et al., 2001).

For classification or regression problems, it has been analytically shown that the gains from using ensemble methods involving weak learners are directly related to the amount of diversity among the individual component models. In fact, for difficult data sets, comparative studies across multiple clustering algorithms typically show much more variability in results than studies comparing the results of weak learners for classification. Thus, there could be a potential for greater gains when using an ensemble for the purpose of improving clustering quality.

The present work, introduces a novel cluster ensemble approach based on boosting, whereby multiple clusterings are sequentially constructed to deal with data points which were found hard to cluster in previous stages. An initial version of this multiple clustering approach was introduced by Frossyniotis et al. (2003). Further developments are presented here including several improvements on the method and experimentation with different data sets exhibiting very promising performance. The key feature of this method relies on the general principle of the boosting classification algorithm (Freund and Schapire, 1996), which proceeds by building weak classifiers using patterns that are increasingly difficult to classify. The very good performance of the boosting method in classification tasks was a motivation to believe that boosting a simple clustering algorithm (weak learner) can lead to a multi-clustering solution with improved performance in terms of robustness and quality of the partitioning. Nevertheless, it must be noted that developing a boosting algorithm for clustering is not a straightforward task, as there exist several issues that must be addressed such as the evaluation of a consensus clustering, since cluster labels are symbolic and, thus, one must also solve a correspondence problem. The proposed method is general and any type of basic clustering algorithm can be used as the weak learner. An important strength of the proposed method is that it can provide clustering solutions of arbitrary shape although it may employ clustering algorithms that provide spherical clusters.

2. The boost-clustering method

We propose a new iterative multiple clustering approach, that we will call *boost-clustering*, which iteratively recycles the training examples providing multiple clusterings and resulting in a common partition. At each iteration, a distribution over the training points is computed and a new training set is constructed using random sampling from the original dataset. Then a basic clustering algorithm is applied to partition the new training set. The final clustering solution is produced by aggregating the obtained partitions using weighted voting, where the weight of each partition is a measure of its quality. The algorithm is summarized below.

Algorithm boost-clustering. Given: Input sequence of N instances $(x_1, \dots, x_N)$, $x_i \in R^d$, $i = 1, \dots, N$, a *basic clustering algorithm*, the number C of clusters to partition the data set and the maximum number of iterations T.

1. Initialize $w_i^1 = 1/N$ for $i = 1, \dots, N$. Set $t = 1$, $\epsilon_{\max} = 0$ and $ic = 0$.
2. Iterate while $t \leqslant T$
   - Produce a bootstrap replicate of the original data set according to the probability w_i^t for every instance i by resampling with replacement from the original data set.
   - Call the *basic clustering algorithm*, to partition the bootstrap training data getting the partition H^t.
   - Get the cluster hypothesis $H_i^t = (h_{i,1}^t, h_{i,2}^t, \dots, h_{i,C}^t)$ for all i, $i = 1, \dots, N$, where $h_{i,j}$ is the membership degree of instance i to cluster j.
   - If $t > 1$, renumber the cluster indexes of H^t according to the highest matching score, given by the fraction of shared instances with the clusters provided by the boost-clustering until now, using H_{ag}^{t-1}.
   - Calculate the pseudoloss:

     $$\epsilon_t = \frac{1}{2} \sum_{i=1}^{N} w_i^t CQ_i^t \tag{1}$$

     where CQ_i^t is a measurement index that is used to evaluate the quality of clustering of an instance x_i for the partition H^t.
   - Set $\beta_t = \frac{1-\epsilon_t}{\epsilon_t}$.
   - Stopping criteria:
     (i) If $\epsilon_t > 0.5$ then
         $T = t - 1$
         go to step 3
     (ii) If $\epsilon_t < \epsilon_{\max}$ then
         $ic = ic + 1$
         if $ic = 3$ then
           $T = t$
           go to step 3
       else
         $ic = 0$
         $\epsilon_{\max} = \epsilon_t$
   - Update distribution W:

     $$w_i^{t+1} = \frac{w_i^t \beta_t^{CQ_i^t}}{Z_t} \tag{2}$$

     where Z_t is a normalization constant such that W^{t+1} is a distribution, i.e., $\sum_{i=1}^{N} w_i^{t+1} = 1$.
   - Compute the aggregate cluster hypothesis:

     $$H_{\text{ag}}^t = \arg \max_{k=1,\dots,C} \sum_{\tau=1}^{t} \left[\frac{\log(\beta_\tau)}{\sum_{j=1}^{t} \log(\beta_j)} h_{i,k}^\tau \right] \tag{3}$$

   - $t := t + 1$
3. Output the number of iterations T and the final cluster hypothesis $H^{\text{f}} = H_{\text{ag}}^T$.

It is clear that the approach has been developed following the steps of the boosting algorithm for classification. We assume a given set X of N d-dimensional instances x_i, a basic clustering algorithm (weak learner) and the required number of clusters C. The maximum number of iterations T of boost-clustering will be considered fixed, although this parameter is meaningless considering the early stopping of the algorithm by the definition of two stopping criteria that we will discuss later on. The clustering obtained at iteration t will be denoted as H^t, while H^t_{ag} will denote the aggregate partitioning obtained using clusterings H^τ for $\tau = 1, \ldots, t$. Consequently, for the final partitioning H^f produced by the clustering ensemble it will hold that $H^f = H^T_{ag}$. The basic feature of the method is that, at each iteration t, a weight w^t_i is computed for each instance x_i such that the higher the weight the more difficult is for x_i to be clustered. In accordance with the boosting methodology, the weight w^t_i constitutes the probability of including x_i in the training set constructed at iteration $t+1$. At the beginning, the weights of all instances are equally initialized, i.e. $w^1_i = 1/N$.

At each iteration $t = 1, \ldots, T$, first a dataset X^t is constructed by sampling from X using the distribution $W^t = \{w^t_i\}$ and then a partitioning result H^t is produced using the basic clustering algorithm on the dataset X^t. For each instance x_i, $i = 1, \ldots, N$, we get a cluster hypothesis $H^t_i = (h^t_{i,1}, h^t_{i,2}, \ldots, h^t_{i,C})$ where $h^t_{i,j}$ denotes the membership degree of instance i to cluster j (we assume that $\sum_{j=1}^{C} h^t_{i,j} = 1$, for all i). It must be emphasized that, although the basic clustering method may be parametric, the boost-clustering method is nonparametric in the sense that the final partitioning is specified in terms of the membership degrees $h_{i,j}$ and not through the specification of some model parameters (e.g. cluster centers). This fact gives the flexibility to define arbitrarily shaped data partitions and makes necessary the use of non-parametric cluster-validity measures, as described in the next section.

In the above methodology the most critical issue to be addressed is how to evaluate the clustering quality of an instance x_i for the partition H^t. We have defined an index CQ^t_i such that the higher the value of CQ^t_i the worse the clustering quality of instance x_i. In our implementation, we considered two ways for computing index CQ.

For the first type, we computed $h^t_{i,\mathrm{good}}$ as the maximum membership degree of x_i to a cluster and $h^t_{i,\mathrm{bad}}$ as the minimum membership degree to a cluster. The CQ index, that will be referred to as *minmax-CQ*, is defined as:

$$CQ^t_i = 1 - h^t_{i,\mathrm{good}} + h^t_{i,\mathrm{bad}} \tag{4}$$

As a second type of CQ index, we propose to use an *entropy-based* measure which takes high values when the membership degree $h^t_{i,j}$ of a data point x_i is comparable for all clusters j, i.e. the point x_i has not been well-clustered:

$$CQ^t_i = -\sum_{j=1}^{C} h^t_{i,j} \log(h^t_{i,j}) \tag{5}$$

Based on the CQ index, at each iteration t the pseudoloss ϵ_t is computed using (1). Then, in analogy with the classification case, the weight distribution w^{t+1}_i for the next iteration is computed using (2). Using this formula, we reduce the weight of a well-clustered data point (i.e. that belongs to a cluster with a high membership degree) and favour the sampling of badly clustered data points. Thus, in analogy with the general principle of the boosting classification algorithm (where specialized classifiers are serially constructed to deal with data points misclassified in previous stages), in each iteration the boost-clustering algorithm clusters data points that were hard to cluster in previous iterations.

A second important issue to be addressed is related to the *cluster correspondence* problem. This means that, in order to define the partition H^t at iteration t, we have to assign an index $l \in \{1, \ldots, C\}$ to each of the C partitions and this indexing must be consistent with those in previous iterations. In particular, we have to decide the one-to-one correspondence between a cluster in partitioning H^t and a cluster in the partition H^{t-1}_{ag}. This correspondence is specified by computing the common patterns between a cluster in H^t and the clusters in H^{t-1}_{ag}. Then, according to the highest matching score, given by the fraction of common samples, the cluster indexes of H^t are renumbered.

In the proposed method, the aggregate clustering result at iteration t is obtained by applying for every instance x_i a weighted voting scheme over the cluster subhypotheses $h^\tau_{i,k}$, $k = 1, \ldots, C$, $\tau = 1, \ldots, t$, using (3), where in analogy with the classification case, the weight of a subhypothesis $h^\tau_{i,k}$ is defined so that a greater weight is assigned to subhypotheses with lower error ϵ_τ. For the early stopping (identification of the optimal value of iterations T) of the algorithm two different stopping criteria were used. In particular, the algorithm terminates if a learner (basic clustering algorithm) has a pseudoloss ϵ_t greater than 1/2 (in which case the partitioning result of the last iteration is not taken into account) or the pseudoloss does not further increase in a number of consecutive iterations (counted by the variable *ic*). Experimentally we found that three iterations is a good choice.

3. Experimental results

In studying the boost-clustering method, we considered two types of basic clustering algorithms, namely the *k*-means and the fuzzy *c*-means algorithm. In our implementation, the membership degree $h_{i,j}$ for every instance x_i to cluster j, for both *k*-means and fuzzy *c*-means, is produced based on the Euclidean distance d

$$h_{i,j} = \frac{1}{\sum_{k=1}^{C} \frac{d(x_i,\mu_j)}{d(x_i,\mu_k)}} \tag{6}$$

where $\mu_j \in R^d$ corresponds to a cluster center. The resulting boost-clustering method using the *min-max-CQ* index (see Eq. (4)) and with *k*-means as the basic clustering algorithm will be referred to as *boost-k-means-minmax* and the one with fuzzy *c*-means as *boost-FCM-minmax*, respectively. Similarly, the resulting boost-clustering method using the entropy-*CQ* index (see Eq. (5)) and with *k*-means as the basic clustering algorithm will be referred to as *boost-k-means-entropy* and the one with fuzzy *c*-means as *boost-FCM-entropy*, respectively. In the experimental study, we compare boost-*k*-means-minmax (or boost-*k*-means-entropy) with the simple *k*-means algorithm and boost-FCM-minmax (or boost-FCM-entropy) with the simple FCM algorithm. To accomplish that, non-parametric cluster-validity measures must be specified as noted in the previous section.

3.1. Non-parametric cluster-validity measures

Since clustering is an unsupervised process and there is no a priori indication for the actual number of clusters present in a dataset, there is need for measures evaluating the quality of a partition. In this spirit, numerous cluster-validity measures have been proposed in the literature. Some of the most commonly used measures are the root-mean-square standard deviation (RMSSTD) and *R*-squared (RS) introduced by Sharma (1996), the intra-over inter-variation quotient, the BD-index introduced by Jain and Dubes (1988) and the SD validity index (Halkidi et al., 2000) which is based on the concepts of average scattering for clusters and total separation between clusters. However, all of these measurements are basically variations on the same theme in that they compare inter-cluster versus intra-cluster variability and tend to favour configurations with bell-shaped well-separated clusters. In our experiments, we considered two non-parametric indices, *isolation* and *connectivity* (Pauwels and Frederix, 1999), which can be used for measuring the validity of *arbitrarily shaped* clusters.

Isolation is measured by the *k*-nearest neighbour norm (NN-norm). In particular, for fixed k (whose specific value is not very critical), the k-nearest neighbour norm $v_k(x)$ of an instance x is defined as the fraction of the k nearest neighbours of x that have the same cluster label as x. A measure of the homogeneity of the total clustering is computed by averaging over all N points in the data set:

$$IS = \frac{1}{N} \sum_x v_k(x) \tag{7}$$

In our experiments we used $k = 0.01N$.

Connectivity relates to the fact that, for any two points in the same cluster, a path should always exist connecting the two points, along which the density of the data remains relatively high. In our implementation, connectivity is quantified as follows: we randomly select K pairs of points (a_i, b_i), $i = 1, \ldots, K$, called anchor-points, such that the points of the same pair belong to the same cluster. Then, for each pair (a_i, b_i), we consider the middle point $\mu_i = (a_i + b_i)/2$ and compute the local density $f(\mu_i)$ by convolving the dataset with a unimodal density-kernel of width σ:

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{2\pi\sigma^2}\right)^{d/2} \mathrm{e}^{-\|x-x_i\|^2/2\sigma^2} \tag{8}$$

Then, the connectivity measure CN (also called C-norm) is computed as

$$CN = \frac{1}{K}\sum_{i=1}^{K} f(\mu_i) \quad (9)$$

In our experiments we chose $K = 0.05N$.

A major drawback of the isolation index, is that it does not notice whenever two clusters are merged, even if they are well-separated (Pauwels and Frederix, 1999). In fact, grouping all samples together in one big cluster, will result in an optimal score for this criterion. For this reason connectivity must be considered as a second criterion to penalize solutions that erroneously lump together widely separated clusters. In order for satisfactory clustering results to be obtained one has to try to maximize both indices simultaneously. However, since there is a trade-off between connectivity and isolation, the two validity indices should be combined to provide a single cluster-validity index, as described in Section 3.3.

3.2. Experimental methodology

In order to demonstrate the performance of the boost-clustering algorithm we considered both artificial and real-world datasets. In the following, we describe the experimental methodology used to compare the boost-k-means-minmax (or boost-k-means-entropy) with the simple k-means algorithm. Note that the same methodology is followed to compare boost-FCM-minmax (or boost-FCM-entropy) with simple FCM. In particular, for each data set and for a specific number of clusters C (the number of clusters for each problem varied from three to six), we applied the following steps:

(1) Split the data set into training and testing set of fixed size.
(2) Run the simple k-means algorithm 20 times, each time with different initialization, to partition the training set into C clusters.
(3) For the 20 runs of k-means compute the values of isolation (IS_{best}) and connectivity (CN_{best}) indexes corresponding to the best of the 20 runs of the k-means, i.e. the one yielding the smallest clustering error on the training set.
(4) Apply the boost-k-means-minmax and the boost-k-means-entropy algorithms on the same training set (using $T = 20$) and compute the corresponding values of isolation (IS_{int} and IS_{entr}) and connectivity (CN_{int} and CN_{entr}) on the test set, respectively.

3.3. Combination of cluster-validity measures

In order to make direct use of the two cluster-validity measures, we compute their Z-scores (Pauwels and Frederix, 1999). The Z-score of an observation ξ_i in a sample $\xi_1, \ldots, \xi_l$ is defined as:

$$Z(\xi_i) = \frac{\xi_i - \text{median}(\xi)}{\text{MAD}(\xi)} \quad (10)$$

where $\xi = \{\xi_1, \ldots, \xi_l\}$ represents the whole sample and MAD stands for *median absolute deviation*:

$$\text{MAD}(\xi) = \text{median}\{|\xi_i - \text{median}(\xi)| : i = 1, \ldots, l\}. \quad (11)$$

Now, let us consider $IS = \{IS_{\text{best}}, IS_{\text{int}}, IS_{\text{entr}}\}$ to be the sample of isolation values and $CN = \{CN_{\text{best}}, CN_{\text{int}}, CN_{\text{entr}}\}$ the sample of connectivity values for the methods we want to compare: (1) best simple clustering algorithm, (2) boost-clustering using the minmax-CQ index and (3) boost-clustering using the entropy-CQ index, respectively. Then, the (robust) Z-score for the ith method is defined as:

$$Z_i = Z(IS\{i\}) + Z(CN\{i\}), \quad i = 1, 2, 3 \quad (12)$$

and we consider as the best clustering result the one which maximizes this robust Z-score on the test set.

3.4. Results and discussion

Four data sets, as shown in Table 1, were used to demonstrate the performance of boost-clustering: the Clouds (see Fig. 1) and Phoneme data sets from the ELENA project (ELENA, 1995), the Page-blocks from the the UCI data set repository (UCI, 1998) and the Banana data set (see Fig. 2) which is an artificial two-dimensional data set consisting of two banana shaped clusters. Table 1 also gives the number of examples used for train-

Table 1
Summary of the data sets

Dataset	Cases	Features		Training set size	Testing set size
		Continuous	Discrete		
Clouds	5000	2	–	2500	2500
Phoneme	5404	5	–	3000	2404
Page-blocks	5473	4	6	3000	2473
Banana	2000	2	–	1000	1000

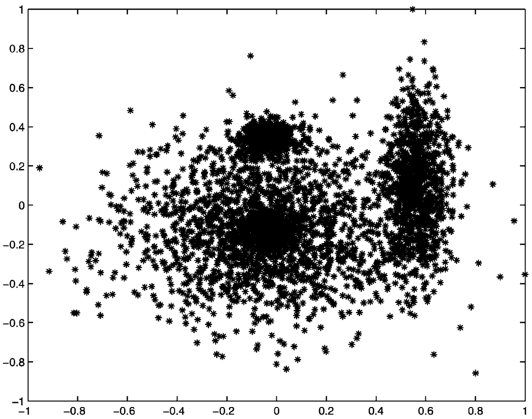

Fig. 1. The Clouds data set.

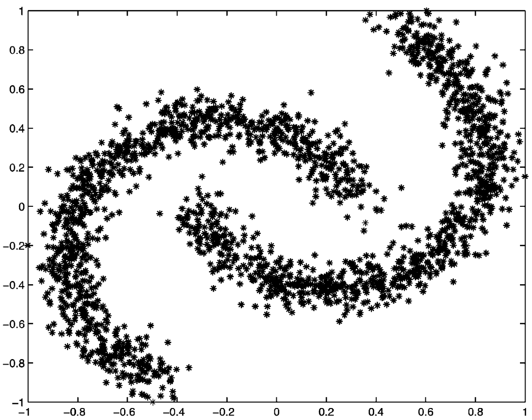

Fig. 2. The Banana data set.

ing and testing respectively, for each data set in the experiments.

Table 2 contains the test set experimental results using the simple k-means and the boost-k-means algorithm. The Z-score (Eq. (12)) index is provided for each of the three compared cases. In each row of Table 2, the best Z-score (maximum value) is highlighted indicating the best partition result. Similarly, Table 3 contains the test set experimental results comparing the simple fuzzy c-means with the boost-FCM algorithm. Running a basic clustering algorithm (such as k-means or FCM) many times, the best one can do is get the best clustering solution, e.g. the one yielding the smallest clustering error on the training set. So, it makes sense to compare the boost-clustering result with the best clustering result produced by many applications of the basic clustering algorithm.

Some experiment examples are displayed only for the boost-k-means-minmax algorithm. Fig. 5 show how the data distribution changes for the Banana data set respectively, after some iterations of boost-k-means-mixmax. The figures on the left hand side show the data points used for training and the figures on the right hand side display the data points that were not sampled in a specific iteration of boost-k-means-mixmax. Examining all figures and considering the general principle of the proposed boost-clustering algorithm, we observe that, as the iterations of boost-k-means-minmax progress, the corresponding training sets contain data points that were hard to cluster in previous iterations. Fig. 3 displays the resulting partition of the Clouds test set in three clusters using the boost-k-means-minmax algorithm and Fig. 4 shows the resulting partition produced from the best simple k-means algorithm. Similarly, Fig. 6 provides the resulting partition of the Banana test set into four clusters using the boost-k-means-minmax algorithm, while Fig. 7 shows the resulting partition produced from the best simple k-means algorithm. It is clear that the boost-k-means-minmax

Table 2
Experimental results for simple k-means and boost-k-means

Dataset	C	Best k-means	Boost-k-means-minmax	Boost-k-means-entropy
Clouds	3	0.501	**0.548**	−1.550
	4	−2.875	**1.605**	1.146
	5	**0.626**	−1.126	0.500
	6	−0.364	0.500	**0.864**
Phoneme	3	**0.233**	−0.162	−0.071
	4	**1.373**	0.500	−0.873
	5	**2.077**	−2.750	0.923
	6	1.445	−3.000	**1.555**
Page-blocks	3	−2.353	**1.331**	0.980
	4	−0.604	0.002	**0.637**
	5	**0.490**	0.000	0.099
	6	−1.232	−0.277	**1.508**
Banana	3	−2.676	−0.394	**2.900**
	4	−1.946	**2.900**	−0.955
	5	−2.500	**2.500**	−0.500
	6	−2.100	0.361	**1.739**

Table 3
Experimental results for simple fuzzy c-means and boost-FCM

Dataset	C	Best FCM	Boost-FCM-minmax	Boost-FCM-entropy
Clouds	3	**2.124**	−3.502	0.373
	4	−2.902	**2.564**	0.336
	5	−1.366	**2.927**	−1.562
	6	0.500	**0.924**	−1.758
Phoneme	3	−1.728	−0.028	**1.757**
	4	**0.707**	−1.207	0.500
	5	−1.602	**2.973**	−1.398
	6	−1.869	**3.100**	−1.030
Page-blocks	3	−2.375	**2.438**	−0.188
	4	−0.104	0.000	**0.174**
	5	−2.044	0.544	**1.000**
	6	−2.929	**3.000**	−1.320
Banana	3	−3.038	**1.970**	0.992
	4	−3.005	**2.509**	0.985
	5	−1.463	**2.929**	−1.608
	6	−0.790	−1.900	**2.890**

algorithm exhibits a better clustering performance for these cases and is able to provide clustering solutions that are not spherically shaped (this is more clear in the Banana dataset), although it employs weak learners providing solutions with spherical clusters.

The experimental results in Table 2 indicate that boost-k-means-minmax outperforms the best k-means in 10 out of 16 experiments. Also, boost-k-means-entropy outperforms the best k-means in 10 out of 16 cases. Overall, the boost-k-means algorithm yielded the best partitioning result in 11 out of 16 cases in comparison with the best k-means which overtopped in five cases. Similarly, the results in Table 3 indicate that boost-FCM-minmax outperforms the best FCM in 13 out of 16 exper-

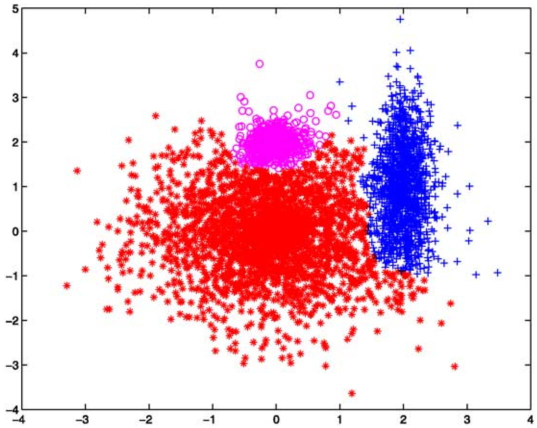

Fig. 3. The resulting partition of Clouds test set into three clusters by the boost-k-means-minmax algorithm.

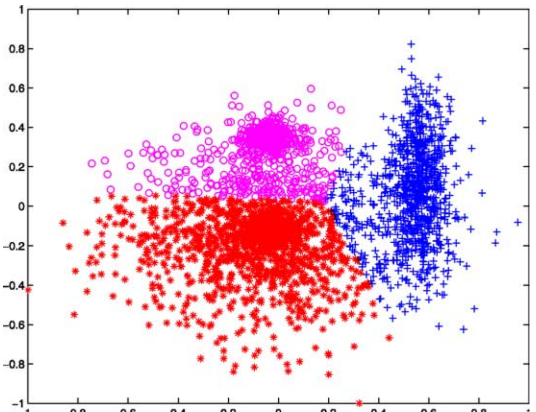

Fig. 4. The best partitioning of Clouds test set into three clusters with the simple k-means algorithm.

iments. Also, boost-FCM-entropy outperforms the best FCM in 11 cases. Overall, the boost-FCM algorithm yielded the best partitioning result in 14 out of 16 cases in comparison with the best FCM which overtopped in only two cases.

The results above indicate that the best way to evaluate the clustering quality index CQ_i of an instance x_i can be very data specific. This suggests that there is no clear reason why the minmax-CQ index is better than the entropy-CQ index. We believe that the entropy-CQ index is more robust and may be beneficial with some large data sets, as in the case of Page-blocks.

An important conclusion that can be drawn is that, in most cases, the boost-clustering algorithm provides better partition results (in 25 of total 32 experiments) than a simple clustering algorithm. Also, we must note here that the boost-clustering algorithm never reached the maximum number of iterations ($T = 20$), in particular the average number of boost-clustering iterations in our experiments was 10. This strongly indicates that boosting a basic clustering algorithm for a small number of iterations can give better results than running the basic clustering algorithm many times and selecting the partition of best run. The performance degradation of boost-clustering, that sometimes occurs when increasing the number of iterations, is in accordance with analogous results reported in the literature concerning the boosting method when applied to classification problems (Freund and Schapire, 1996; Wickramaratna et al., 2001). In the case of the boost-clustering algorithm, overtraining may be observed after many iterations, especially for data sets with noisy patterns or outliers which are hard to cluster. Another reason for performance degradation is the distortion of the overall structure of the original data set due to resampling, hence the early stopping criterion of boost-clustering is critical.

A critical issue relevant to the effectiveness of the proposed method concerns the problem of inferring the optimal number of clusters. Indeed, there is no well-established method to describe the structure of arbitrary shaped clusters, as defined by the proposed boost-clustering algorithm. It is well known that the k-means (or fuzzy c-means) algorithm, based on a minimum square error criterium, identifies hyper-spherical clusters, spread around prototype vectors representing cluster centers. Techniques for selecting the number of clusters according to this optimality criterium basically identify an optimal number of cluster centers on the data that splits it into the same number of hyper-spherical clusters. When the data exhibits clusters with arbitrary shape, this type of decomposition is not always satisfactory. Inadequate data partitions, such as the partitions plotted in Figs. 4 and 7, can be obtained even when the correct number of clusters is known a priori. These

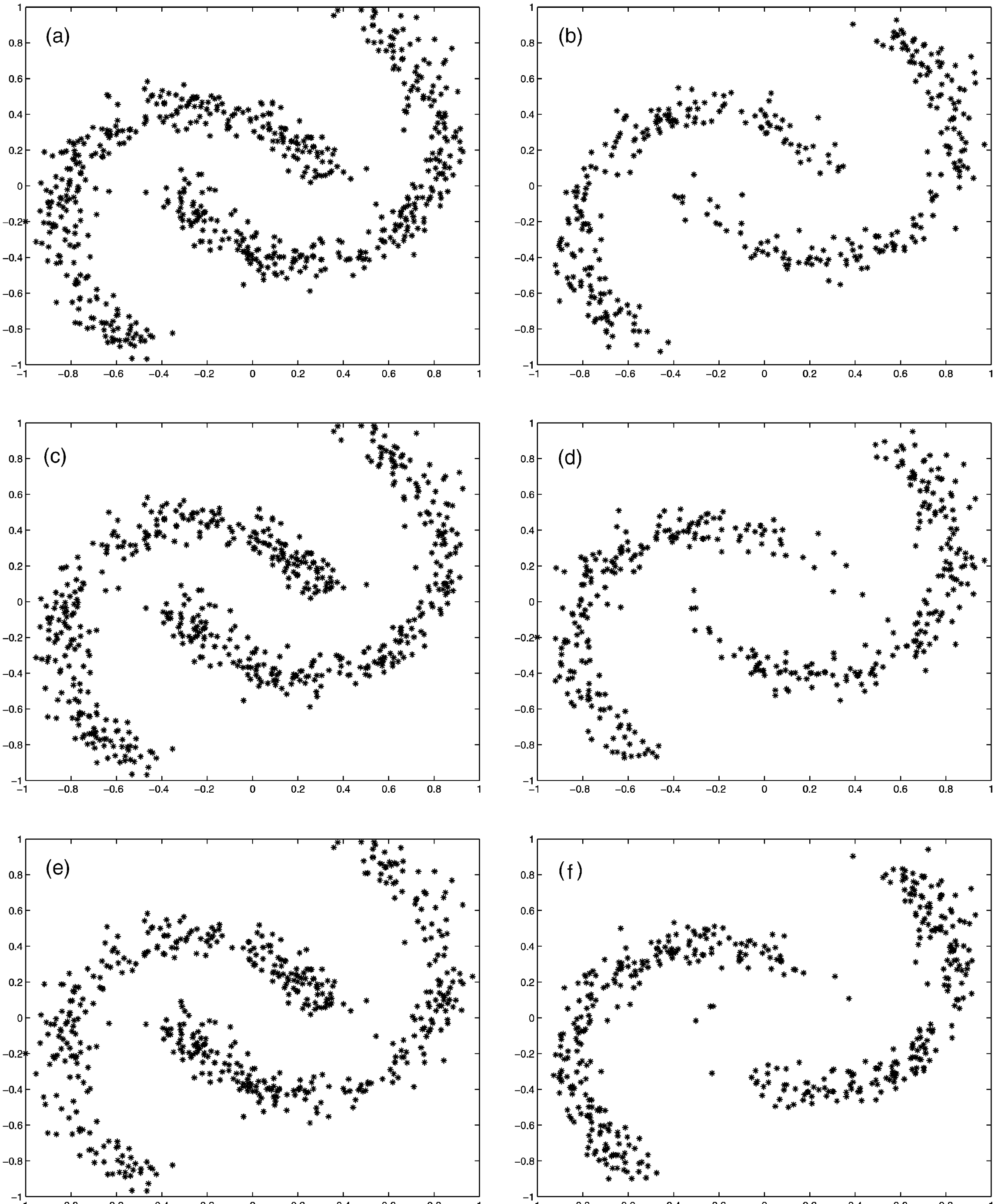

Fig. 5. Banana data set: The resampled data points used for training (a, c and e) and the data points that were not sampled (b, d and f) by the boost-k-means-minmax algorithm (with $C = 4$), after 2, 4 and 7 iterations respectively. (a) Sampled data points after two iterations. (b) Non-sampled data points after two iterations. (c) Sampled data points after four iterations. (d) Non-sampled data points after four iterations. (e) Sampled data points after seven iterations. (f) Non-sampled data points after seven iterations.

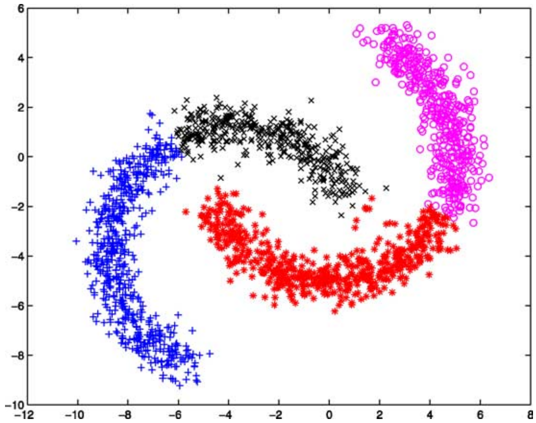

Fig. 6. The resulting partition of Banana test set into four clusters by the boost-k-means-minmax algorithm.

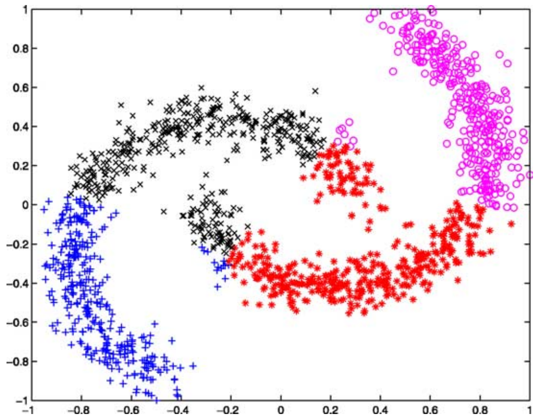

Fig. 7. The best partitioning of Banana test set into four clusters with the simple k-means algorithm.

misclassifications of patterns are however overcome by using the proposed boost-clustering methodology, setting C to the known number of clusters.

The complexity and scaling behaviour of the proposed clustering approach depends on the complexity of the basic clustering algorithm since we have set an upper bound on the number of boosting iterations. For example, the complexity of k-means is $O(kn)$, where k is the number of clusters and n is the size of the dataset. It must be noted that in none of the experiments the algorithm reached this upper bound.

3.5. Comparative results with a Bagging clustering approach

Both the Bagging (Breiman, 1994) and boosting techniques are competitive in classification tasks, so it could be interesting to test the comparative performance of the proposed boost-clustering technique with respect to a Bagging clustering approach. Therefore, we implemented another ensemble clustering approach (Bagg-clustering) inspired from Bagging. Actually, the method is similar to the boost-clustering algorithm, with a major difference being that at each iteration the sampling distribution is uniform for all data points. The resulting Bagg-clustering method with k-means as the basic clustering algorithm will be referred to as *Bagg-k-means* and the one with fuzzy c-means as *Bagg-FCM*, respectively.

The data set used in our experiments is two-dimensional with 4000 patterns, containing several arbitrarily shaped clusters, as shown in Fig. 8. We followed the same experimental methodology using 2000 patterns for training and 2000 for testing, considering the two non-parametric indices, *isolation* and *connectivity*, for measuring the validity of arbitrarily shaped clusters. Table 4 contains the test set experimental results using the simple k-means, the boost-k-means and the Bagg-k-means algorithm. Similarly, Table 5 contains the test set experimental results comparing the simple fuzzy c-means with the boost-FCM and the Bagg-FCM algorithm. We must note here that, in each

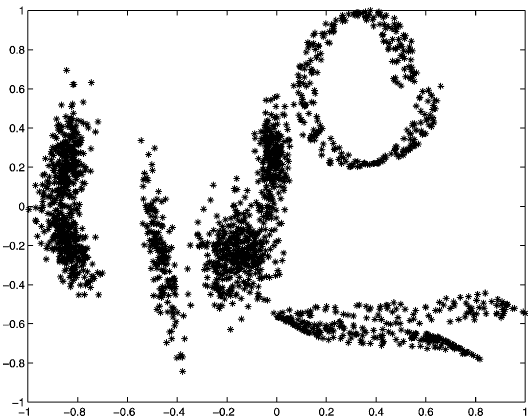

Fig. 8. The data set that we used in our experiments.

Table 4
Experimental results for simple *k*-means, boost-*k*-means and Bagg-*k*-means

C	Best *k*-means	Boost-*k*-means-minmax	Boost-*k*-means-entropy	Bagg-*k*-means
5	−1.617	**5.000**	−1.648	−0.734
6	−1.768	**1.200**	0.550	−0.233
7	−0.870	−0.141	**1.120**	−0.108
8	−3.500	1.534	**1.910**	0.055

Table 5
Experimental results for simple fuzzy *c*-means, boost-FCM and Bagg-FCM

C	Best FCM	Boost-FCM-minmax	Boost-FCM-entropy	Bagg-FCM
5	−0.144	−0.989	**0.655**	−0.190
6	−2.333	**1.901**	1.566	−2.000
7	−1.134	0.250	**1.045**	−0.160
8	−2.348	0.656	**2.508**	−0.566

case, the Bagg-clustering algorithm was run for the same number of iterations as boost-clustering. Fig. 9 displays the resulting partition of the data set in seven clusters using the boost-*k*-means-entropy algorithm and Fig. 10 shows the resulting partition produced from the best simple *k*-means algorithm (the lines indicate the cluster borders).

Although, the experimental results indicate that Bagg-clustering outperforms the best simple clustering algorithm in 7 out of 8 experiments, overall, the boost-clustering algorithm yielded the best partitioning result in all the cases. The key future behind the good performance of Bagg-clustering is the reduction of variability in the partitioning results via averaging, but, as in the classification case, since boosting uses a more efficient way of sampling the data, it provides better clustering solutions.

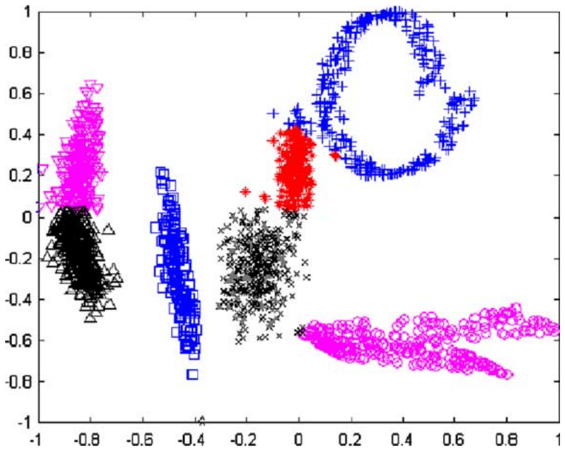

Fig. 9. The resulting partition of the data set into seven clusters by the boost-*k*-means-entropy algorithm.

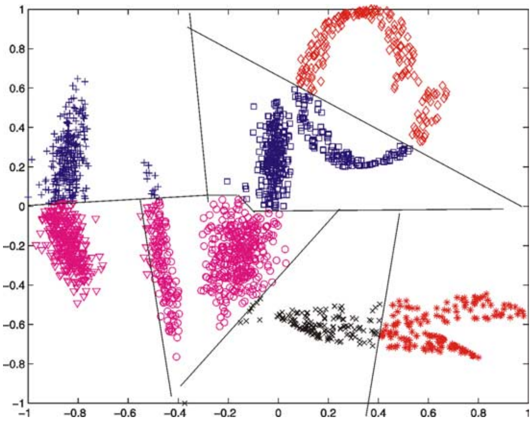

Fig. 10. The best partitioning of the data set into seven clusters with the simple *k*-means algorithm.

4. Conclusions

In this work a new clustering methodology has been introduced based on the principle of boosting. The proposed method is a multiple clustering method based on the iterative application of a basic clustering algorithm and the appropriate aggregation of the multiple clustering results through weighted voting. The proposed algorithm treats the problem of local minima of clustering algorithms by iteratively running a basic clustering algorithm, however its performance is not influenced by the randomness of initialization or by the specific type of the basic clustering algorithm used. In addition, it has the great advantage of providing clustering solutions of arbitrary shape though using weak learning algorithms that provide spherical clusters, such as the *k*-means. For the quality measurement of data partitioning we considered a cluster-validity index resulting from the combination of two non-parametric indices, *isolation* and *connectivity*.

We conducted experiments using several data sets, to illustrate that boost-clustering can lead to improved quality and robustness of performance. The method is very promising, as the experimental study has shown that boosting a basic clustering algorithm (even for a few iterations) can lead to better clustering results compared to the best solution obtained from several independent runs of the basic algorithm. We have also carried out an experimental study comparing the proposed boost-clustering algorithm with a Bagging clustering approach. Both ensemble methods lead to improvements in clustering performance, but, in general, the adaptive resampling scheme used in boost-clustering provides better solutions compared to Bagging.

There exist several directions for future work with the boost-clustering algorithm. The most important direction is to determine the optimal number of clusters existing in the data set. The desired number of clusters is often not known in advance. In fact, the right number of clusters in a data-set often depends on the scale at which the data is inspected, and sometimes equally valid (but substantially different) answers can be obtained for the same data. Moreover, the selection of the optimal number of clusters can be incorporated in the clustering procedure, either using local or global cluster validity criteria. On the other hand, in order to determine an adequate value or range of C, one should use some a priori information (for instance, by applying a mixture decomposition method for determining the number of components in the mixture). Otherwise, several values of C should be tested, the final number of clusters being the most stable solution found. Other interesting future research issues concern the specification of alternative ways for evaluating how well a data point has been clustered, as well as the experimentation with other types of basic clustering algorithms.

References

Bezdek, J., 1981. Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York.

Bezdek, J., Pal, S., 1992. Fuzzy Models for Pattern Recognition: Methods that Search for Structures in Data. IEEE CS Press.

Boundaillier, E., Hebrail, G., 1998. Interactive interpretation of hierarchical clustering. Intell. Data Anal. 2 (3).

Breiman, L., 1994. Bagging predictors. Tech. Rep. 421, Department of Statistics, University of California, Berkeley.

Dempster, A., Laird, N., Rubin, D., 1977. Maximum likelihood from incomplete data via the em algorithm. Roy. Statist. Soc. B 39, 1–38.

ELENA, 1995. ESPRIT Basic Research Project ELENA (no. 6891). Available from <ftp://ftp.dice.ucl.ac.be/pub/neural-nets/ELENA/databases>.

Fisher, D., 1987. Knowledge acquisition via incremental conceptual clustering. Machine Learn. 2, 139–172.

Fred, A., 2001. Finding consistent clusters in data partitions. In: Proceedings of the Second International Workshop on Multiple Classifier Systems (MCS 2001) Lecture Notes in Computer Science, 2096. Springer, Cambridge, UK, pp. 309–318.

Freund, Y., Schapire, R., 1996. Experiments with a new boosting algorithm. In: Proceedings of the Thirteenth International Conference on Machine Learning. Bari, Italy, pp. 148–156.

Frossyniotis, D., Likas, A., Stafylopatis, A., 2003. A boosting approach to clustering. In: Proceedings of Joint International Conference on Artificial Neural Networks and Neural Information Processing (ICANN/ICONIP), June 26–29, 2003, Istanbul, Turkey. pp. 196–199.

Frossyniotis, D., Pertselakis, M., Stafylopatis, A., 2002. A multi clustering fusion algorithm. In: Proceedings of the Second Hellenic Conference on Artificial Intelligence, April 11–12 LNAI 2308. Springer-Verlag, Thessaloniki, Greece, pp. 225–236.

Halkidi, M., Batistakis, Y., Vazirgiannis, M., 2001. Clustering algorithms and validity measures. In: Proceedings of the 13th International Conference on Scientific and Statistical Database Management, July 18–20. IEEE Computer Society, George Mason University, Fairfax, Virginia, USA.

Halkidi, M., Vazirgiannis, M., Batistakis, L., 2000. Quality scheme assessment in the clustering process. In: Proceedings of the 4th European conference on Principles of Data Mining and Knowledge Discovery Lecture Notes in Computer Science, vol. 1910. Springer, Lyon, France.

Jain, A., Dubes, R., 1988. Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs, NJ.

Jain, A., Duin, R., Mao, J., 2000. Statistical pattern recognition: A review. IEEE Trans. Pattern Anal. Machine Intell. 22 (1), 4–37.

Pauwels, E., Frederix, G., 1999. Finding salient regions in images: Non parametric clustering for image segmentation and grouping. Comput. Vision Image Understand. 75, 73–85.

Sharma, S., 1996. Applied Multivariate Techniques. John Willey & Sons.

Smyth, P., 1996. Clustering using Monte Carlo cross-validation. In: Proceedings Knowledge Discovery and Data Mining, pp. 126–133.

UCI, 1998. UCI Machine Learning Databases Repository. University of California-Irvine, Department of Information and Computer Science. Available from <ftp://ftp.ics.edu/pub/machine-learning-databases>.

Vlassis, N., Likas, A., 2002. A greedy-EM algorithm for Gaussian mixture learning. Neural Process. Lett. 15, 77–87.

Wickramaratna, J., Holden, S., Buxton, B., 2001. Performance degradation in boosting. In: Proceedings of the 2nd International Workshop on Multiple Classifier Systems MCS2001, vol. 2096. Springer, pp. 11–21.